Prajjwol Tripathi

Microclimate Research Module for Nests

TICC2650 Sensortag and IBM Cloud Integrated System

Helsinki Metropolia University of Applied Sciences

Bachelors of Engineering

Electronics Engineering

Thesis

Date 8/5/2018

Helsinki Metropolia University of Applied Sciences

Author(s) Title Number of Pages Date	Prajjwol Tripathi Microclimate Research Module for Nests 40 pages + 2 appendices 8th May 2018
Degree	Bachelors of Engineering
Degree Programme	Electronics Engineering
Instructor(s)	Matti Fischer (Principal Lecturer)

Goal of this project was to study Texas Instrument's CC2650 Sensor Tag Module and Cloud Computing environment of IBM Cloud. A detailed study of these two subjects was done, as well as a Cloud Sensor product was designed.

Motive behind this thesis was to study about the possibilities of using SensorTag module to study birds nesting behaviour by collecting various sensor data and extracting those data through cloud. Need for implementing Texas Instruments product was realised, as few of the earlier prototypes that were designed by other fellow students, were not ideal for the existing situation. Previously a similar Arduino Senor Bundle was created to understand bird's nesting behaviour, but its physical complexities and lack of user friendliness prevented it from being used in a real nesting situation. So, a fast easy and user-friendly system was on constant search when we were introduced to a SensorTag Module designed by Texas Instruments.

Ultralow powered Texas Instrument's Simplelink CC2650 Sensor Tag is a pocket-sized sensor module that consists 10 different varieties of sensors embedded with the latest and powerful ARM Cortex M3 Processer. Sensor data can be collected immediately as the device starts with a simple user friendly mobile application connected over bluetooth.

By integrating SensorTag with IBM Cloud, a cloud server environment was created, where incoming data from sensor's were stored in an IBM Cloud database, which can later be extracted to physical database, according to the need. This thesis can also be used as a basic guideline for students and professionals who are new or unfamiliar to electronics world and embedded systems. These could be students who want to play with sensors and create a weather station of their own or professionals who need a real-time data sets to work on their Big Data Research's.

Keywords	IBM Cloud, SensorTag, IOT, NodeRed

Contents

Acronyms

AI	Artificial Intelligence
API	Application Programming Interface
BLE	Bluetooth Low Energy
CCS	Code Composer Studio
GAP	Generic Attribute Profile
HCI	Host Controller Interface
IAAS	Infrastructure as a Service
IOT	Internet of Things
IDE	Integrated Development Environment
IBM	International Business Machine
JSON	JavaScript Object Notation
MQTT	Message Queuing Telemetry Transport
MCU	Microcontroller Unit
PAAS	Platform as a Service
PCB	Printed Circuit Board
ROM	Read Only Memory
SAAS	Software as a Service
SPI	Serial Peripheral Interface
TCP	Transmission Control Protocol
TI	Texas Instruments
UART	Universal Asynchronous Receiver-Transmitter

1 Introduction

IOT is taking over world by storm. Everything is under a constant surveillance of sensor systems, from smart home to smart nests. Data flow from these sensors are immense. Data generated by these sensor is helping us to make smarter decisions in every step by letting us foresee future as we go.

The aim of this thesis was to create a simple user-friendly sensor controlled system for ornithologists to study bird's nesting behaviour. A sensor module was designed by integrating TI CC2650 SensorTag and IBM Cloud which are one of the most user-friendly sensor device and cloud platform available around us.

About ornithologist's need for this device, there are mostly five different period of nesting cycle researchers are interested in: egg laying, incubation, hatching, feeding the young and leaving the nest. Each period can tell us about the evolution of species and change of behaviour with change in the surrounding can tell a lot about change in whole of our ecology. Some of the earlier research came to an understanding that no matter how the outside weather is changing, parent birds keep a well-regulated temperature during the incubation and brooding process, as little birds grow [1]. To study these behaviours and many other, a quick easy to setup system is always a necessity for both hobbyists and researchers.

Initially, SensorTag was tested with mobile as a reader. Further study introduced its easy cloud connectivity, which led us to use mobile phone as a gateway. As our project aimed to create a system that reads data constantly and maintain a continuous device connectivity, one of the power saving feature of SensorTag prevented us from doing so. A TI Debugger was bought to re-program its firmware to disable its sleep mode functionality and finally an IBM Cloud system was created to constantly update collected sensor data and a real-time dashboard to view current device status.

This thesis also aims to introduce to public about sensors and cloud systems and act as a standalone document to begin with their own personal sensor project with SensorTag and IBM Cloud.

2 Sensors

Sensors have slowly become an inseparable part of our everyday life. Like living organisms use their senses to survive, sensors give sense to technology as whole to perceive, analyse and respond. Sensors have unlocked possibilities to new era of AI Humanoids and other advanced technologies.

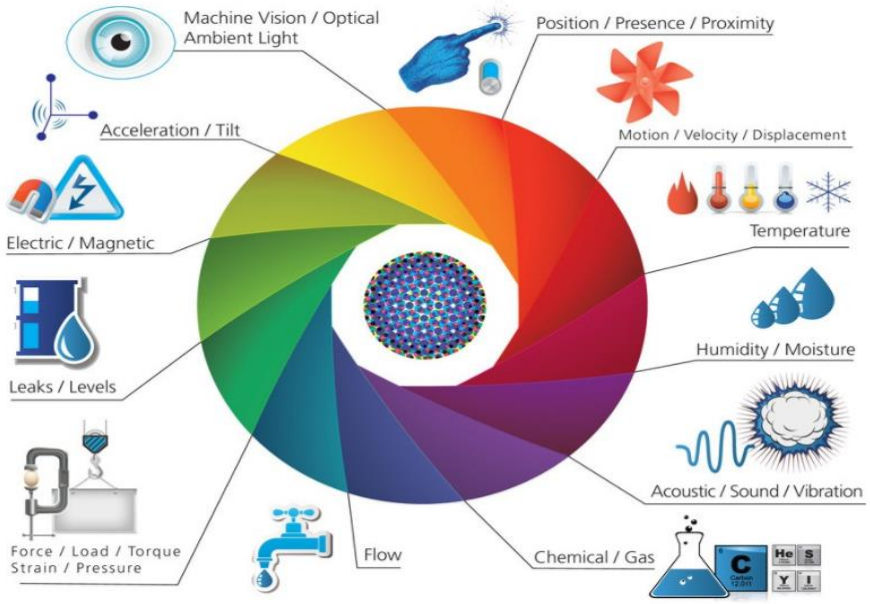

Figure 1 Sensors and Actuators [24]

Figure 1 shows some of the examples of sensors that are being used and their measured stimulus. For example, Machine Vision used in robotics and security locks uses optical ambient light to detect retinas, accelerometers and gyroscopes are used to detect change of speed and tilted motions of objects and humidity sensors to detect air moisture levels. [2]

According to American National Standards Institute, sensors can be defined as a device that responds to specified measurements relating to a useable output. Sensors convert physical phenomena to electrical signal. Transducer which convert one form of energy to another is the heart of any sensor system. Some of the commonly detectable phenomena are:

- Biological and Chemical
- Optical

- Mechanical
- Magnetic
- Thermal and Electric

Some of the commonly measured quantities of these stimulus are shown in Table 1 .

Table 1 Stimulus and their measured quantities [2]

Stimulus	Measured Quantities
Biological and Chemical	Fluidity
Optical	Reflection, Refraction, Absorption
Mechanical	Position, Velocity, Force, Pressure, Torque, Strain, Stress
Thermal	Temperature, Flux, Heat
Electric	Voltage, Current, Electric Field, Conductivity, Permittivity
Magnetic	Magnetic Field, Flux

3 Internet of Things (IoT)

“In just 6 short minutes, you can learn how to connect Raspberry PI using Node-RED and Watson IoT Platform to bring a dinosaur to life. IoT really is becoming the Internet of Everything “ [3] . IBM application qurator Michelle L Corbin describes about IoT and its future implications in one of her blogs.

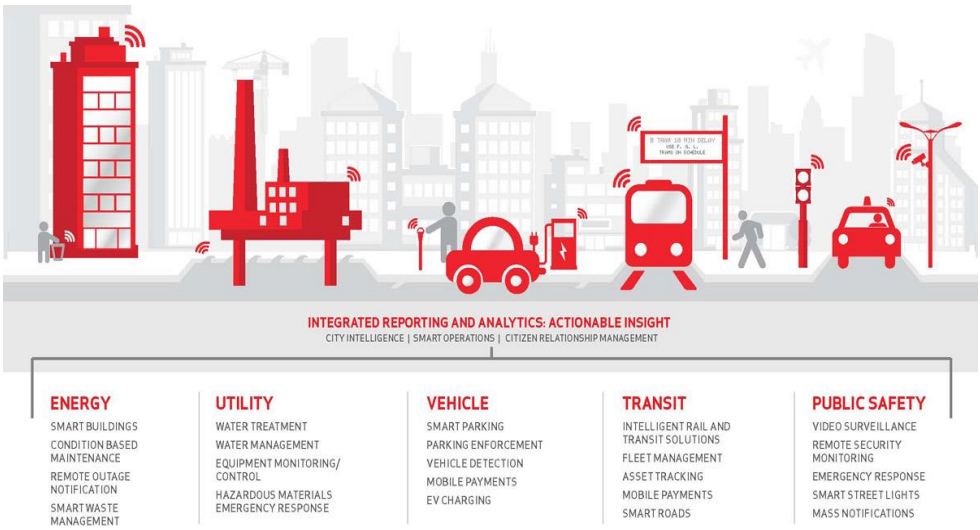

Figure 2 IOT for Smart City Concept [26]

Term “Internet of Things” was coined by Keven Ashton in 1999. IoT can be defined in many ways, according to McKinsey “Sensors and Actuators embedded in physical objects are linked through wired and wireless networks often using the same internet protocol (IP) that connects the internet” [4] .Smart city concept like the one in Figure 2 excites lot of investors and engineers. IoT being a new topic in world of technology its service market possibilities are endless from water cargos to home water supplies.

A classic example of IoT world would be your main gate sensing your car, your car being parked inside garage on its own. Your smart home will know you are home and will open the door and switch on the lights on its own, you might want to drink a coffee once you are home and your coffee machine start, your tv could switch you to your favorite contents, you might fall asleep after some time and your home will go to sleep mode as well. IoT in near future will make it possible for humans to avoid everyday mechanical unproductive tasks. Some of IoT services already in the market are shown in Figure 3 .

Figure 3 IoT devices in market [25]

Devices like Activity Trackers are already making a lot of impact on people’s everyday life. Not only fitness conscious people are utilizing its benefit but also devices like these are helping people to focus on their health status and motivate them to exercise and improve their exercising habits.

4 Wireless Sensor Networks in Wildlife

“Theory of natural selection” by Charles Darwin describes his work based on observation of Galapagos birds, which has changed the way we understand the evolution of not only humans, but also other animals [5]. Wireless Sensor networks are nowadays used widely by researchers in the field of biology and ecology to study behavior and

changes in living organisms. Unlike satellite imaging systems used earlier, use of sensor networks have given researchers a close-up insight to everyday life of animal kingdom. As we have all realized by now, animal kingdom, including humans, is an interlinked biological network that influences the planet's entire ecosystem. There are mainly two types of sensor network systems based upon ways sensor nodes are deployed.

4.1 Static Network

Sensor nodes are placed in places where studied creatures are specially attached to, places like bird's nests, river banks and burrows, hot spots of static network system [1].

4.2 Dynamic Network

Dynamic Networks are sensor nodes that are attached to creatures under observation. Examples are birds with radio frequency identification tags that will notify when a migrated bird returns to a territory, tagged rhinos and tigers in national parks are also examples of dynamic network system [1].

Figure 4 Study of duck burrows with Sensor Networks [6]

An example of Static Network in Figure 4 shows how Ornithologists were using Sensor Networks to study burrows back in early 2000.

1. Storm Petrel birds on Maine's Great Duck Island were studied by sensors inside their burrows.
2. With a radio module nailed outside.
3. Data recorded from the sensors were relayed to the gateway node.
4. These data were then transmitted to the desktops in nearby lab.
5. Then a satellite dish in the labs rooftop would transmit data to the primary research lab in California [6]

5 Simplelink CC2650 SensorTag

TI CC2650 is National Instruments newest sensor tag MCU coming from CC26** family, an upgraded, cost-effective and ultra-low powered module. Smaller in size, as can be seen from Figure 5 and with a capacity to run on a coin sized battery, its applications and future possibilities are immense.

Figure 5 Simplelink CC2650 SensorTag [7]

A unique ultra-low power sensor controller embedded in ARM Cortex M3 Processor creates an ideal environment for collecting various analog and digital data [7] .
Some of the most notable features of this device are [7]:

- Powerful Arm Cortex M3 Processor
- Instant IBM Cloud Connectivity
- Sensor controllers can run autonomously from rest of the system
- Bluetooth Low Energy (BLE)

- Programmable Current Source
- True Random Number Generator
- Real Time Clock
- Wide Supply Voltage Range
- 2.4 GHz RF Transceiver
- Extremely Flexible for IOT Applications

Its instant data visualisation feature is one of the most attractive part of Texas Instruments SensorTag devices which is making this device popular in science classes. Robust and power efficient Bluetooth solutions have made this device popular among researchers and designers.

5.1 System Architecture

Main CPU is powered by ARM Cortex M3 Processor. Its exceptional computational rate in addition to memory use and power consumption make it very suitable for systems with least data implementation [8].

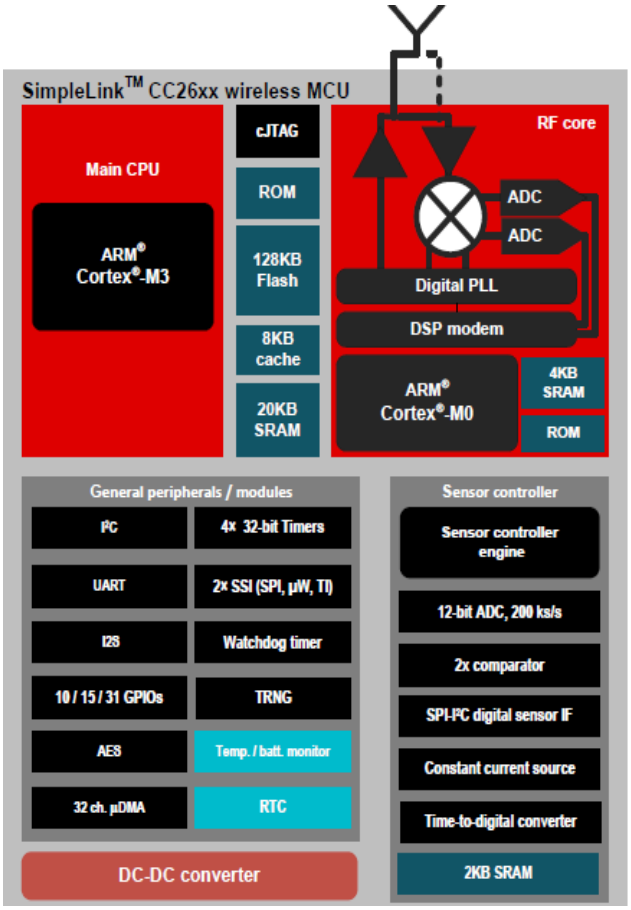

Figure 6 Functional Block Diagram [8]

RF Core contains an ARM Cortex M0 Processor which only takes care of baseband circuitries. Since this unit is working independently handling all the radio protocols a substantial amount of processing load is taken off the main systems back. Radio protocols that are handled by the system are 802.15.4 RF4CE, ZigBee and BLE. [7]

Sensor Controller Unit shown in the bottom right section of the block diagram in Figure 6 plays vital role in optimising power consumption by performing task autonomously. Hence, main CPU no longer needs to be active while this module is at work.
This unit is configured in such a way that just the right level of processing amount is needed for sampling sensor data and making sensor decisions which, in turn, makes it more power efficient. [7]

General Peripheral Unit's I2C Interface can communicate between other systems that follow I2C standards. UART implements a universal asynchronous receiver/transmitter function. SSIs are synchronous serial interfaces that are compatible with SPI, MICROWIRE, and Texas Instruments synchronous serial interfaces. Unit's Static RAM is divided into four memory blocks: two 4 kilobytes and two 6 kilobytes. They are mainly used for data storage and code execution. ROM holds a bootloader that allows device to be reprogrammed through SPI or UART interface. [8]

5.2 Hardware Configuration

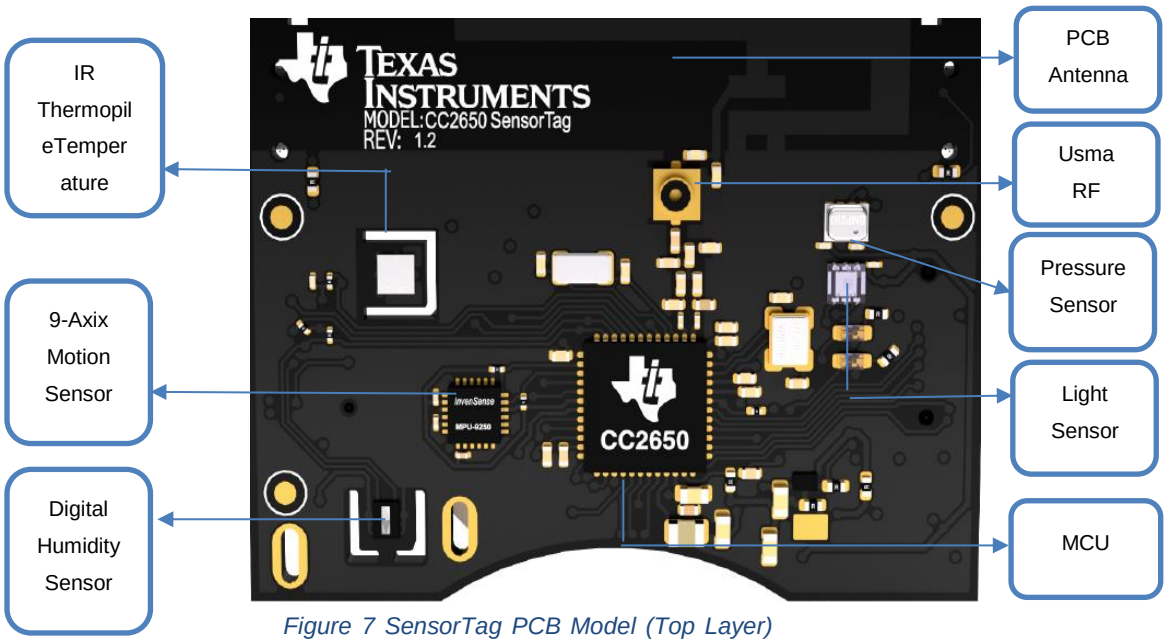

Figure 7 SensorTag PCB Model (Top Layer) [8]

Understanding the hardware configuration opens more possibilities to solve problems with better results. Shown below in Figure 7 and Figure 51 (Appendix 1) is a PCB model of (top and bottom layer) of SensorTag.

Compact size and on the go usability has made this device quite popular among makers. Some of the hardware feature are shown below in Table 2. CC2650 version of SensorTag has 10 sensors that can be used at once in a matter of seconds.

Table 2 Sensors and their Datasheet Reference Name [2]

Sensor	**Name**	**Manufacturer**
Optical	OPT3001	Texas Instruments
Pressure	BMP280	Bosch
Humidity	HDC1000YPA	Texas Instruments
Movement	MPU9250	InvenSense
IR Temperature	TMP007	Texas Instruments
Digital Microphone	SPH0641LU4H	Knowles Electronics

Movement Sensor MPU9250 is also known as 9-Axis Motion Sensor Device as it embeds 3-Axis Magnetometer, 3-Axis Accelerometer and 3-Axis Gyroscope in a single package. These sensors use I2C communication protocol. SensorTag also carries high end radio module with a separate ARM Cortex M0 microcontroller unit. It supports three radio protocols BLE, ZigBee and 6LoWPAN all of which uses 2.4 GHz carrier frequency and an inverted F PCB patch antenna used as a transceiver. [9]

5.3 SensorTag Sensors

Sensor's that are active and collecting data in this project is described below for better understanding of its behavior.

5.3.1 Optical Sensor

CC2650 SensorTag uses OPT3001 Ambient Light Sensor which claims to have a significant infrared rejection with an optical filtering capability closest to human eye. Figure 8 shows Spectral Response comparison of OPT3001 and Human Eye [8].

A healthy human eye can respond to 390 nm to 700 nm of wavelengths [10]. OPT3001 has a measurement accuracy of 0.01 k lux to 83 k lux and a body size of 2 mm * 2 mm and has an operating temperature range of -40 ^{0}C to 85 ^{0}C. OPT3001 in SensorTag can be configured by re writing source code named *sensortag_opt.c* if need arises [7].

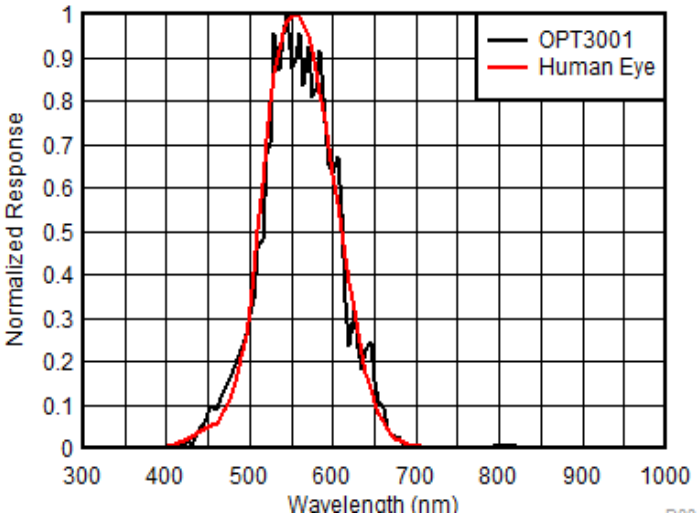

Figure 8 Spectral Response of OPT30001 and Human Eye [8]

5.3.2 Humidity Sensor

HDC1000 is a digital humidity sensor with temperature sensing capabilities. It is designed to function under very low power with high measurement accuracy which makes its use ideal for low power IoT projects. -40 ^{0}C to +125z ^{0}C functional temperature range further makes its use reliable in harsh climatic conditions which in our case is nests and burrows. Source code used in CC2650 SensorTag for HDC100 can be edited from file sensortag_hum.c.

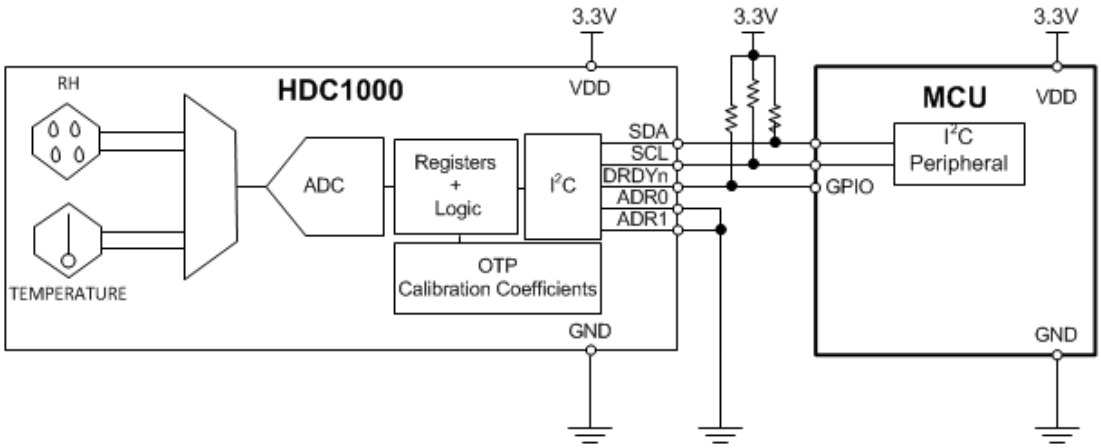

Figure 9 HDC1000 with Microcontroller Unit [8]

A typical model of HDC1000 interfaced with microcontroller is shown in Figure 9. Here both temperature and humidity of the surrounding is measured by the sensor which later can be controlled and monitored from MCU. [7]

5.3.3 Barometric Pressure Sensor

BMP280 from Bosch Sensortec is used in SensorTag for Barometric Pressure Sensing. Specially designed for battery driven wearables due to its compact size it alone can measure digital humidity, pressure and temperature.

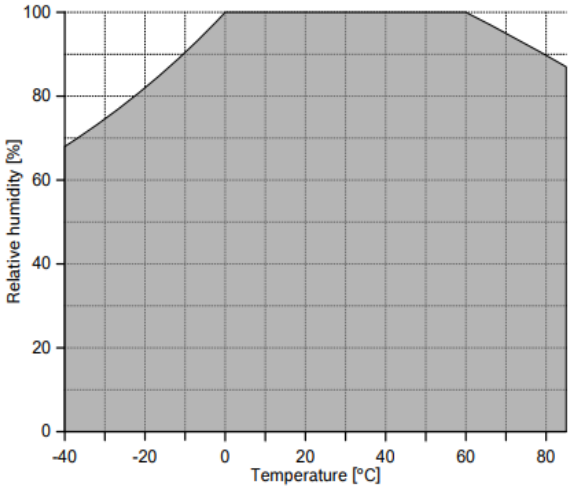

Figure 10 BMP280 Sensor Operation Range [7]

Figure 10 shows BMP280 operates with full measurement accuracy from 0^0C to +60^0C [7] . Barometric Pressure is also known as atmospheric pressure and its fluctuations are the first signs of changing weather conditions.

5.4 SensorTag Network Standards

CC2650 SensorTag follows three different radio protocols Bluetooth Low Energy (BLE), ZigBee, and 6LowPAN for remote control applications. [11]

To work with each of these radio modules, separate firmware should be loaded to MCU

- http://www.ti.com/tool/ble-stack has firmware for BLE Compliant Devices
- http://www.ti.com/tool/z-stack contains SensorTag ZigBee firmware to download.

- https://github.com/contiki-os/contiki 6LowPAN firmware for SensorTag can be downloaded from this link.

5.5 ARM Cortex M3

ARM Cortex M3 is a 32-bit processor designed especially for System on Chip (SoC) real time applications. Some of the key features and applications of ARM Cortex M3 are: [12]

- Efficient interrupt handling
- Efficient debug and development features
- Reduced Pin Count Requirement
- Interrupts automatically save/restore state
- Supports Multiple Sleep Modes

Ideally suited for small electronics products from toys to sensor modules and due to its Low Interrupt Latency and high-performance efficiency, it is ideal for real time automotive systems [13].

5.6 Bluetooth Low Energy

BLE is a wireless protocol for battery powered low energy devices that requires low bandwidth to communicate. A new profile architecture and new protocol stack separates BLE from traditional Bluetooth.

Table 3 shows general specifications of BLE [14]:

Table 3 BLE General Specification [14]

Range	50 m open field
Output Power	10 mW (10dBm)
Max Current	15 mA
Latency	3 ms
Topology	Star
Connection	billions
Modulation	GFSK 2.4 GHz

Security	128-bit AES CCM
Sleep Current	1µA
Modes	Broadcast, Read, Write, Connection

BLE Protocol stack, architecture shown in Figure 11, controls all aspects of communication between Controller and Host and is composed of two main parts: Controller and Host Controller consists of two layers physical layer and link layer. Host runs on application processor and controls device connectivity, discoverability and security. [14]

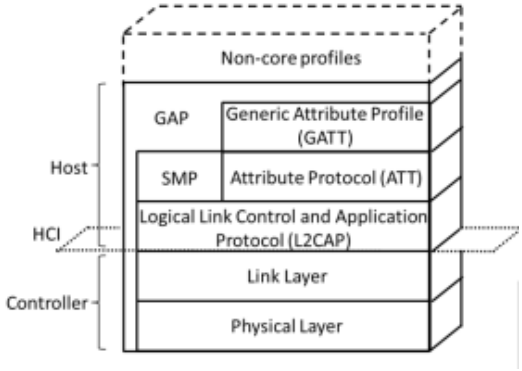

Figure 11 BLE Protocol Stack [2]

To Debug SensorTag as to make it advertise indefinitely, we are interested in GAP of Host layer which controls device connectivity and discoverability [14]. In our case we are interested in GAP Protocol of our peripheral device which is our SensorTag.

5.7 Live Graphs and Real Time 3D Visualisation

One of the most notable feature of SensorTag is its ability to visualise graphical data with its mobile application with just a click. Figure 12 is accelerometers graphical data that can be directly visualised from the mobile application.

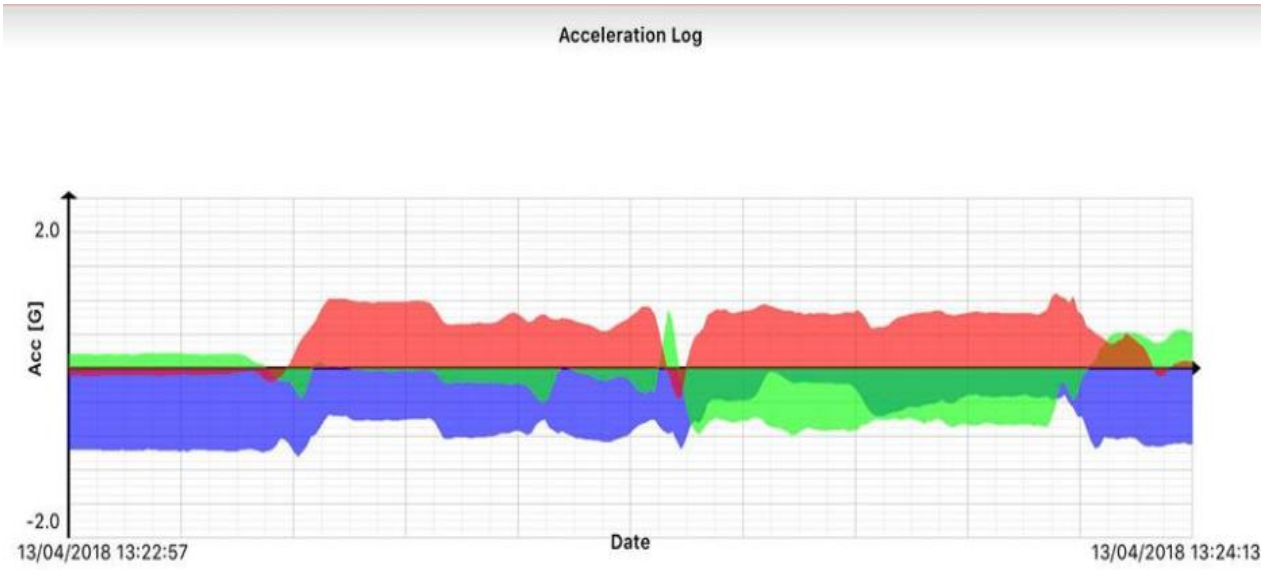

Figure 12 Acceleration (XYZ) graph from iOS application

User can also interact with this 3D model by tapping in. We can remove its covers as we tap in the device as shown in Figure 13 and explore its hardware model without having to remove its physical covers. This service is available only in iOS applications now.

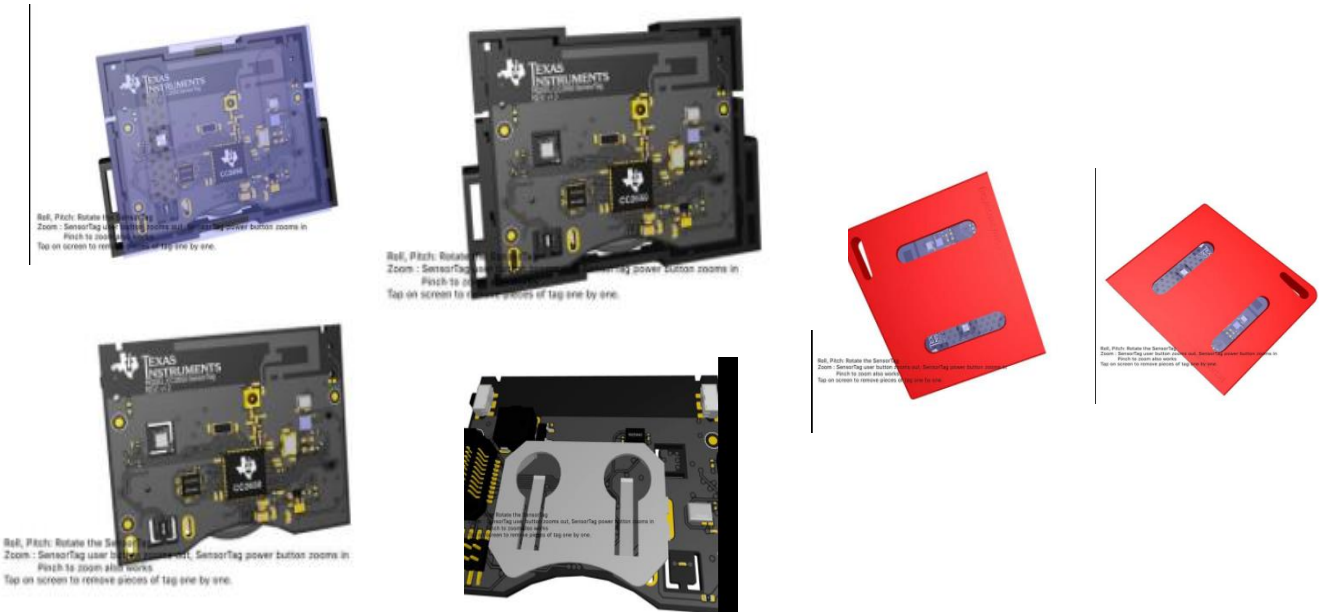

Figure 13 Interactive live 3D model

6 IBM Cloud

IBM Cloud gives end user necessities such as privacy and data protection, data processing and data storage as a service over internet. It provides computing in overall sense as a service so user can only concentrate on precise task they intend to fulfil, rest of the backend services requiring expertise is taken care by the cloud itself.

IBM Cloud is an all in one hub of cloud services that small to big companies can use to effectively deploy their product. IBM Cloud formerly known as Bluemix provides public, dedicated, and local integrated deployment models [15]. All the resources deployed through IBM Cloud are hosted by the IBM Data Centre of user's choice that are linked globally. Some of the key features of IBM Cloud are [14].

- High computational processing service which in other case could need high amount of hardware investment.
- High storage facility
- Flexibility to test range of applications and use of open sources gives extra wings to creators.
- Data Security
- Pay per use service makes its use more popular among small companies.

6.1.1 Software as a Service (SAAS)

End users can directly use existing applications in Cloud. This way they can completely avoid installation and software update issues. Figure 14 Software as Service (SAAS) refers to the similar idea. [15]

Figure 14 Software as Service (SAAS) [15]

6.1.2 Platform as a Service (PAAS)

Create and deploy web based products immediately and start collecting market data without the cost of requiring hardware and server necessities. Git-Hub is one of the popular cloud platform where users can share write and deploy codes over the internet, one similar concept is illustrated in Figure 15 [15].

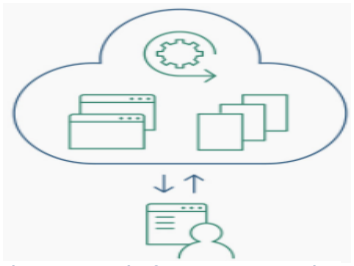

Figure 15 Platform as a service [15]

6.1.3 Infrastructure as a Service (IAAS)

Access all the server hosting and data storage facilities on pay per use basics which could help smaller business to concentrate on product development than to engage upon data security issues or hardware infrastructure problems illustrated in Figure 16. Cloud storage facilities that are widely available is a good example of IAAS where service providers are allowing end users to use their hard drives. [15]

Figure 16 Infrastructure as a Service [15]

6.1.4 Watson IoT Platform Starter

Watson IoT Platform Starter is where our work takes place. Two of the necessary elements that this project require are Sensor Tag's connection to IBM cloud and database to store real time data, which are both available to use under IBM cloud Lite plan for free.

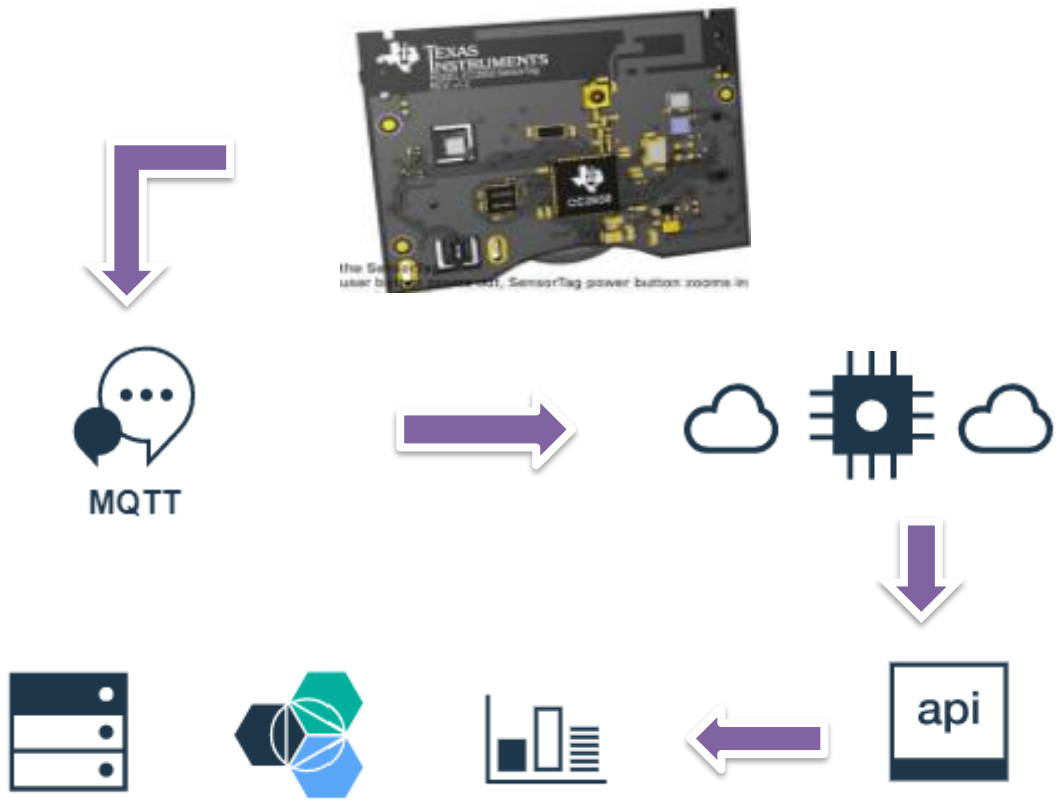

Figure 17 IBM IoT Architecture [15]

Figure 17 IBM IoT Architecture takes us closer to understanding IoT and specially how IBM IoT functions. A device or a gateway is connected to IBM Watson, MQTT messaging protocol creates a safe passage between device and IBM IoT. Once connected we can read, analyse and manipulate data to get meaningful insights. To make further interactive applications, built in API's can be used to create mobile or web applications [16].

6.1.5 NoSQL IBM Database

Cloudant NoSQL DB is a database designed to operate with modern web applications and mobile applications that can be easily accessed with a URL. They are fully managed by IBM Cloud and are free to use (a gigabyte [15]) under IBM Lite Plan. Cloudant makes it easy to conduct advanced analytics on JSON data with dash DB Warehousing and Apache Spark integrations [15]. In this project we are using Cloudant NoSQL Database to store our live data when the gateway is active to cloud storage and make it readable from NodeRed web interface.

7 NodeRed Programming

NodeRed is growing popularity among IoT Community due to its simplicity and effectiveness. NodeRed is also referred as visual tool for building Internet of Things as shown in Figure 18. We can imagine devices connected over a network being virtually wired to create a real-time system, which IoT is all about [17].

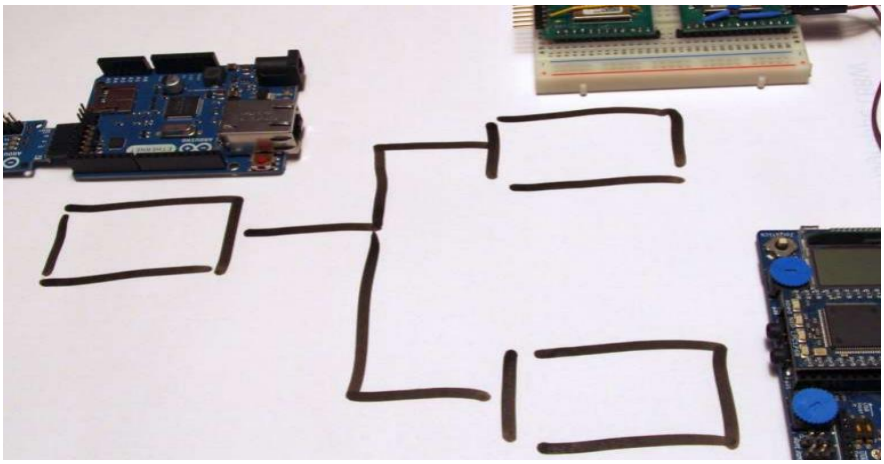

Figure 18 Visualising NodeRed Programming Concept [17]

Easy to use, short learning curve, rapid prototyping and its abilities to be integrated with the latest IoT devices is making it popular among IoT Developers. NodeRed can be run locally and can be integrated with devices like Raspberry Pie and Arduino or in the cloud. NodeRed architecture is shown below in Figure 19.

NodeRed servers can be run locally by installing Node.js but in our context since we are using NodeRed through IBM Cloud NodeRed servers are managed by IBM Cloud itself, so we can easily execute our flows through the browser without configuring server.

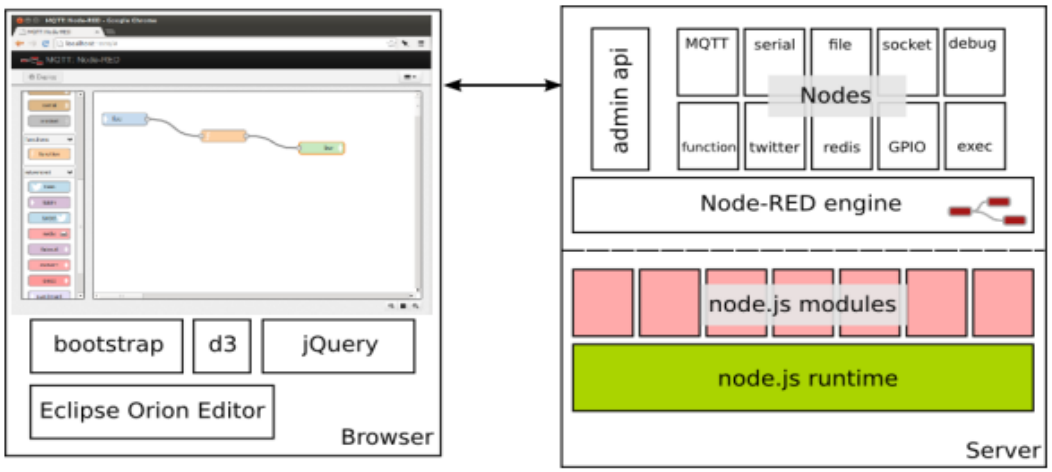

Figure 19 NodeRed Architecture [17]

7.1 Node.js

Node.js is an event driven networking engine written in JavaScript and NodeRed is lightweight runtime built upon Node.js. Node.js is mostly used for building event driven server-side applications.

7.2 NodeRed User Interface

NodeRed gathers all the stream of events both physical and digital into a single window that makes up Internet of Things. Predefined code blocks are nodes. Series of nodes are interconnected with each other creates a flow (Figure 20).

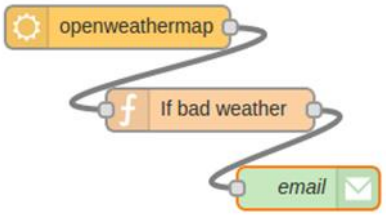

Figure 20 NodeRed Flow of Events [17]

Node Red Flow editor can be accessed easily if an IOT system is created with IBM Cloud and can also be used locally to program devices like raspberry pi, configuring node red through node.js [17].

7.2.1 NodeRed Panel

NodeRed UI which is used over internet through IBM Cloud is shown in Figure 21.

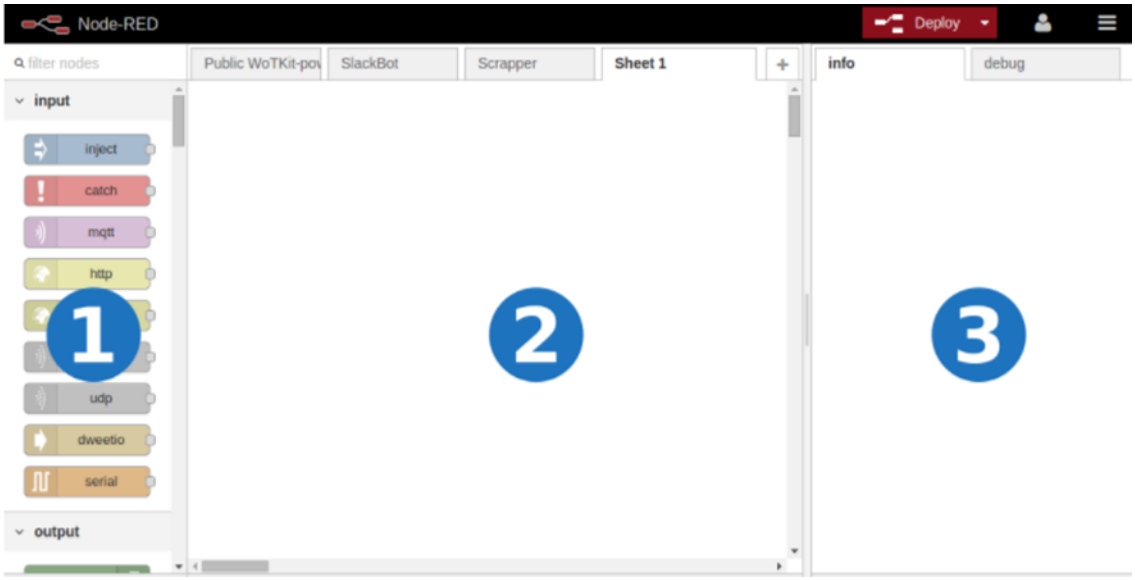

Figure 21 NodeRed UI [17]

1. NodeRed Panel: Organised in categories (input, output, functions, IOT, Social, Dashboard) Node Red Panel consists of lists of basic nodes.

2. Sheet Panel: Sheet Panel is where all the node red flows are created it's the main working space. Nodes from node red panel is drag and dropped into sheet panel and are interconnected accordingly to form flows.

3. Info and Debug Panel
   Info panel shows information about a selected node about what that node means and how it can be used. Information printed with the debug node is seen in debug panel, errors that occurs with flows can also be seen in debug panel.

7.3 Nodes

Each node takes information, processes it and gives necessary output and messages from nodes are in JSON format. There are three types of basic nodes:

- Input Node
- Output Node
- Process Node

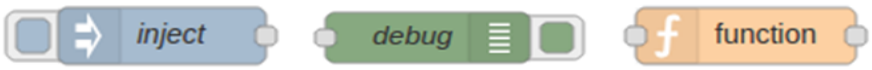

Figure 22 Function Node [17]

Input nodes (blue node in Figure 22) are used to input data to flow editor from other services such as Social Networks, web sockets or TCP. Information can be fed in manually into flow using inject node e.g. a random even number can be fed into system every second as a simulator using inject node [17].

Output nodes (green node in Figure 22) sends data to other services such as IBM Cloud, TCP, MQTT messages or to debug frame [17].

Process node (pink node in Figure 22) processes data works as a function, processed data is then passed to another node inflow or saved in database for future reference, it can also be used as functions to trigger alerts or delays [17].

7.3.1 Configuring Nodes

Each node is unique but can be configured easily by reading information from the info tab from info and debug panel when nodes are selected shown in Figure 23 [17].

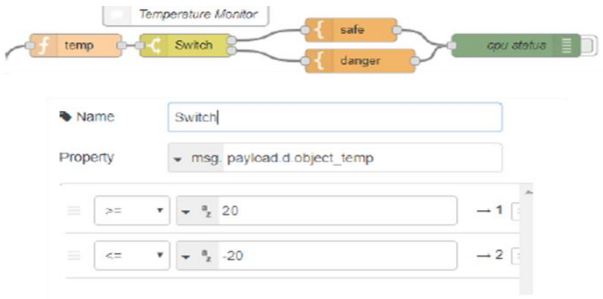

Figure 23 Nodes Configuration [17]

7.4 Deploying

After the flow is ready we can deploy by simply clicking the deploy button.

8 Application Development

This is where all the above information is applied and needed, a real-time sensor cloud environment was designed to read and record all the sensor data over time in a cloud database (Figure 24). Data such as temperature, humidity and infrared temperature tell a lot about birds nesting time and these data can help researchers to analyse behavioural habits of birds during that season.

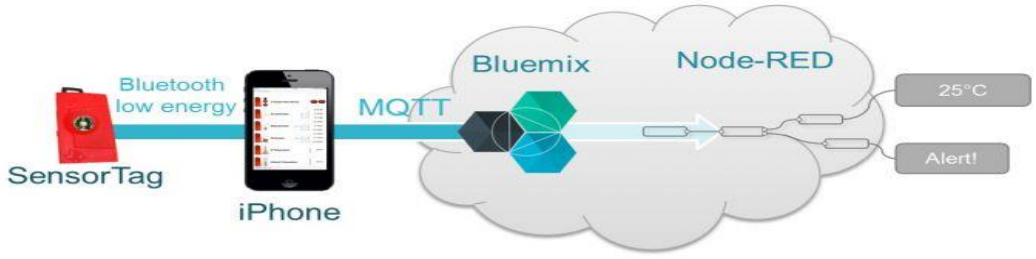

Figure 24 Microclimate Module Architecture [27]

8.1 SensorTag Configuration

There are three basic steps to be followed to get SensorTag up and running which can be followed from Appendix 1 (SensorTag Configuration).

On device start up, SensorTag goes for a self-test of its sensors and firmware, green led blinks rapidly on its success. There are two buttons on each side, power button and indicator button. Power button, when pressed for three seconds, disconnects SensorTag and indicator button, when pressed only indicates the button status. On the contrary, when two buttons are pressed together for six seconds, factory settings are restored. This especially means something, when device is loaded with ZigBee image and user wants to return to its original BLE Operation. [7]

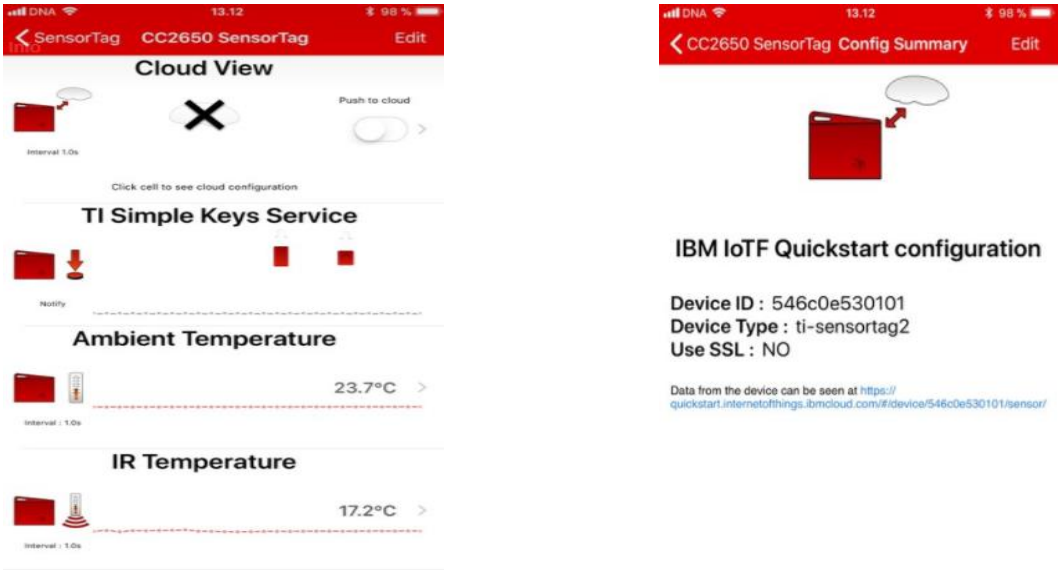

Figure 25 Sensor data view (right) and push to cloud feature (left)

Once the device is up and running live sensor data can be visualised through IBM Cloud. Push to cloud feature can be enabled by turning on Push to Cloud Slider on the upper right corner of mobile application in Figure 25. Once doing so Device Type and Device ID of the SensorTag is given to the user. Live data can now be visualised by simply pressing the link given in Figure 26 i.e. https://quickstart.internetofthings.ibmcloud.com/#/

Once the device ID is correctly given to the Quickstart page, live data can be visualised immediately, as can be seen in Figure 26.

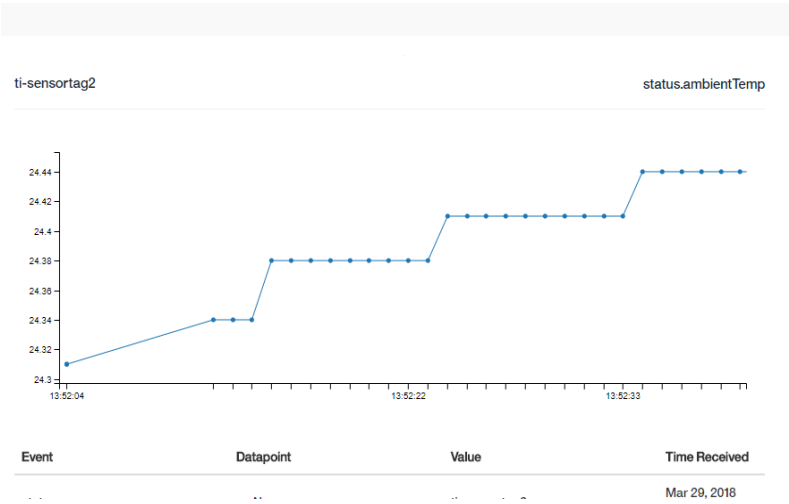

Figure 26 Live Data from SensorTag to IBM Cloud

8.2 Setting up IBM Cloud

User can see IBM Dashboard after successfully creating a free IBM Cloud account.

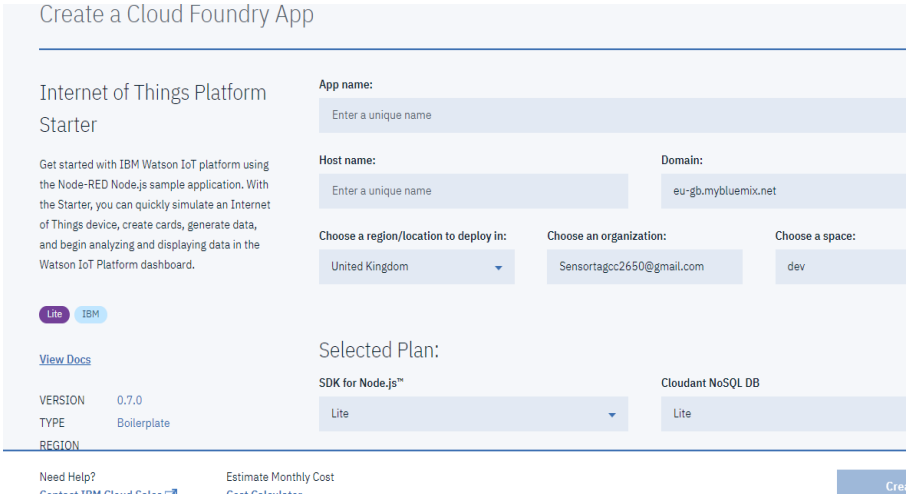

Figure 27 Creating a Cloud Foundry App

Dashboard is where it all begins. It let's user to create resources and user can access all the features available, but since we are using a free account, only lite plans are available to use, which is all we need for our project.

Internet of Things Platform starter is where our work takes place since it contains three basic features our application needs as shown in Figure 27.

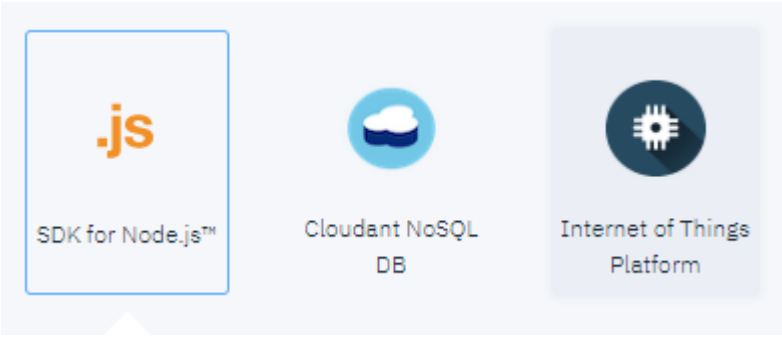

Figure 28 Services provided by Internet of Things Platform Starter

Now we can create our own IOT Cloud Foundry Application. Application name and a unique host name should be provided (Figure 28) and since we are not investing in our application, Lite Plans are selected for all the services which are free but with limited data capacity, which is good enough for our microclimate module.

Here our application name is **microclimate** which is shown below in Figure 29

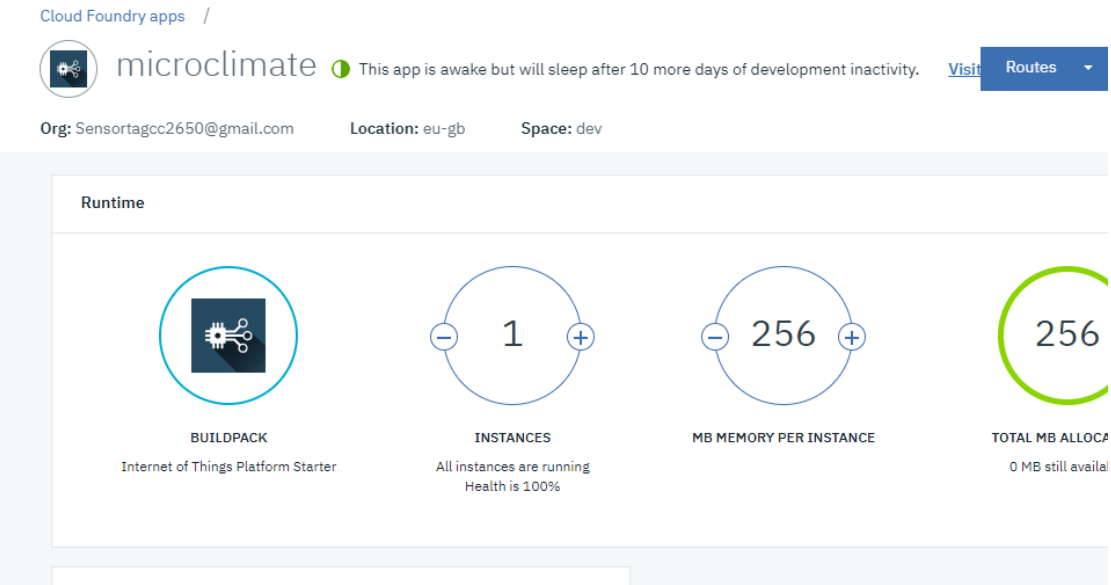

Figure 29 Application Dashboard

Since we are using NodeRed as a connecting backbone of our device, a separate NodeRed workspace should be opened to switch workspace which can be simple accessed by clicking the Visit URL on the upper right corner of Figure 29.
Similarly, Watson IOT Platform can be opened in parallel to create a live systematic graphical dashboard.

8.3 Integrating IBM Cloud and SensorTag

We are visualising our Sensor data with two different platforms with a NodeRed live data dashboard and Watson IOT Platform. NodeRed is used here to connect SensorTag and IBM Cloud with a live database to store sensor data. Watson IOT Platform creates a real-time data visualisation dashboard and explore other possibilities to use in our system, such as data analytics.

8.3.1 Creating a flow with NodeRed

Once the NodeRed opens with above mentioned procedure in 8.2 and Figure 29, a username and password needs to be authorised to restrict access to NodeRed. Before programming it is needed to get to know some basics of NodeRed from this document section NodeRed Programming. Once familiar with some basics, we can start programming. Figure 21 shows the NodeRed User Interface.

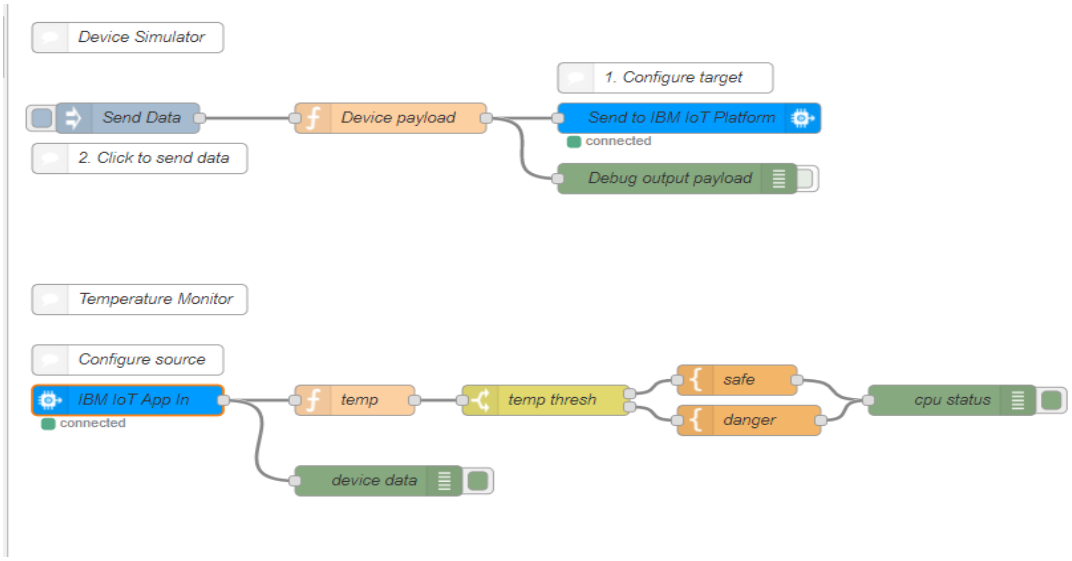

Figure 30 Demo flow from IBM Cloud

When NodeRed is connected to IBM Cloud for the first time, IBM Cloud creates a test flow to show some basic functions of NodeRed. Figure 30 shows two flows being executed: upper one *Device Simulator* section simulates a random temperature values and displays in a dashboard through IBM IoT Platform where we can see live data, *Temperature Monitor* section takes the simulator values and checks if it exceeds a safe threshold [18].

So, before writing our own flow, we can play around with this demo program for a while to understand how functions and conditions work in NodeRed.

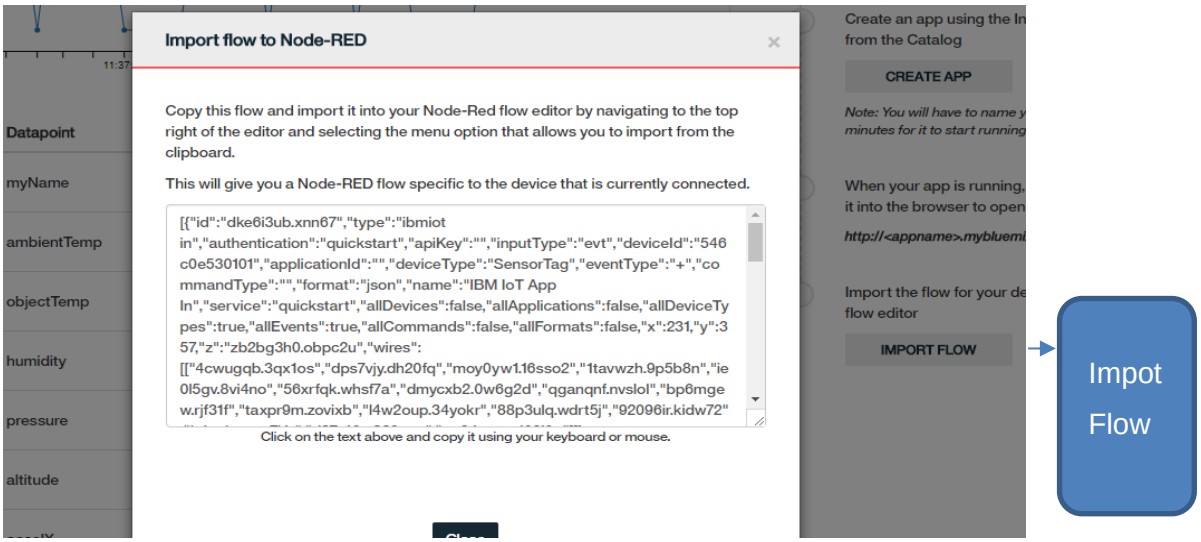

Figure 31 Device Specific Flows

IBM Quickstart page let's user import device specific flows for NodeRed in Figure 31 which we should copy and import to the clipboard. Input node category has a built-in node called ibmiot which connects IoT applications to IBM Cloud.

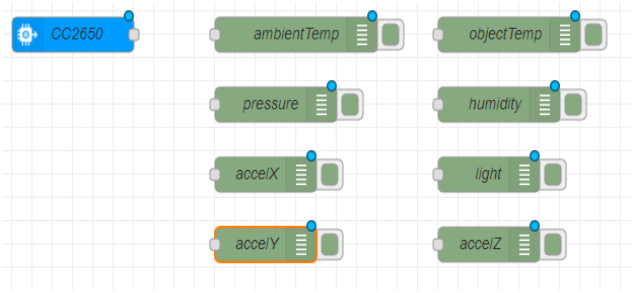

Figure 32 Imported Flow

We can authenticate our SensorTag by using Device ID. In this case imported flows automatically generates device input node and output nodes as shown in Figure 32.In Figure 32 we are using only 9 sensors so rest is removed from the flow, now to create functions and conditions we need to know each sensor property names by clicking sensor nodes property, which can be seen in Figure 33.

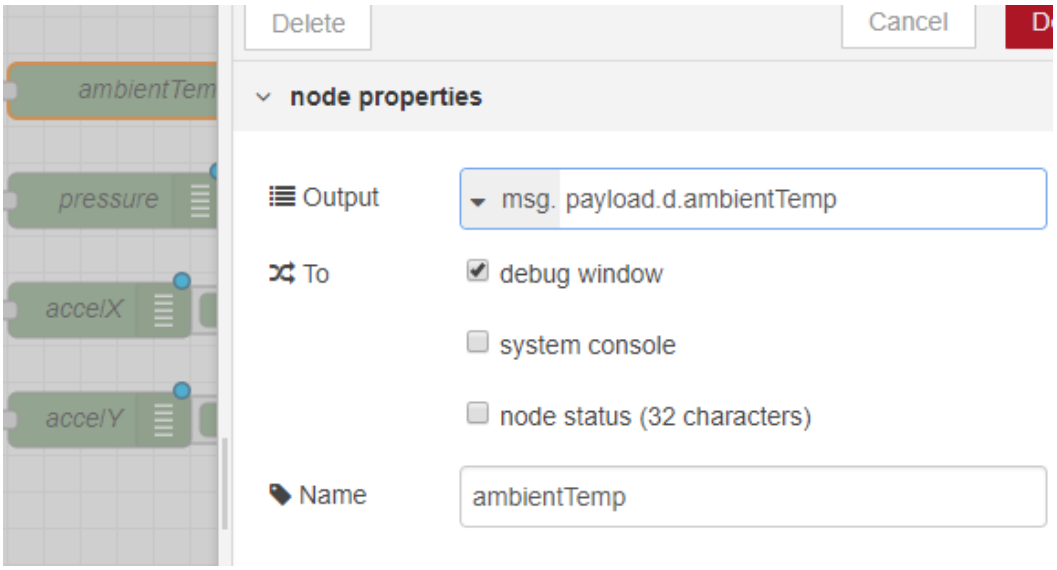

Figure 33 Sensor Node Properties

We can simply test the live sensor data in debug panel of Node Red once we connect all the nodes.

In Figure 34 we can see how connection is made between SensorTag and sensor nodes. Here debug panel updates data every minute since a delay node is used. Once we successfully test our sensor data we can now program it to create a Cloudant NoSQL Database. In Figure 35 live data from SensorTag is sent to Cloud Database node which represents Cloudant NoSQL Database by IBM Cloud.

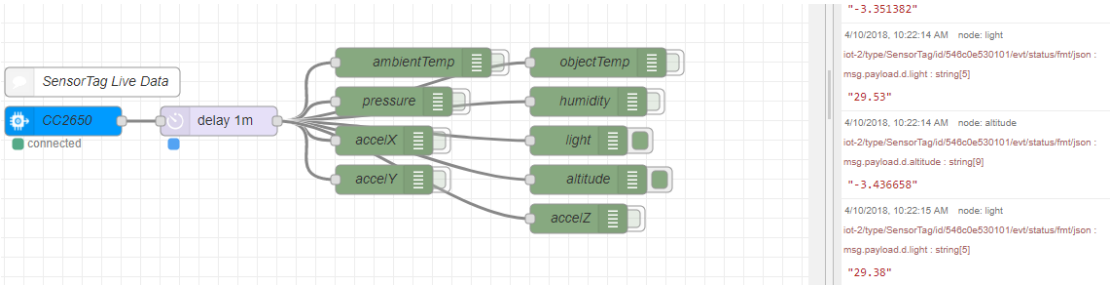

Figure 34 Testing Live Sensor Data

Limited Sensors function node first extracts specified sensor values and Indexes those numeric data to its specified string value and uploads it to Cloudant Database is code can be found in Appendix 2. Data are uploaded every 15 minutes to cloud by using a delay node.

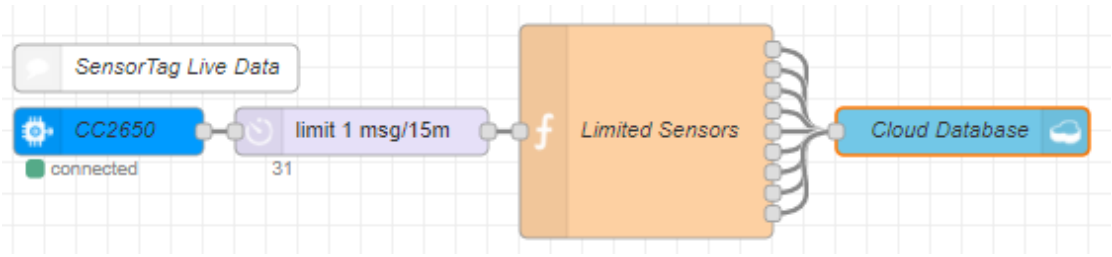

Figure 35 SensorTag to cloud database

Once we successfully upload our data to cloud, we can check it by going to our cloud database platform described in 6.1.5 We need an easy access to our live database, going through the IBM platform could be bit hectic if we need to check uploaded data every now and then, so we will create a HTTP Platform which will extract our data from the cloud server and displays it to a web page https://microclimate.eu-gb.mybluemix.net/sensortag .

In Figure 36 HTTP Nodes establishes a HTTP Platform over the NodeRed Server, then requests Database for data which is then converted into a Table by using tableify node (tableify node converts JSON objects into HTTP Table).

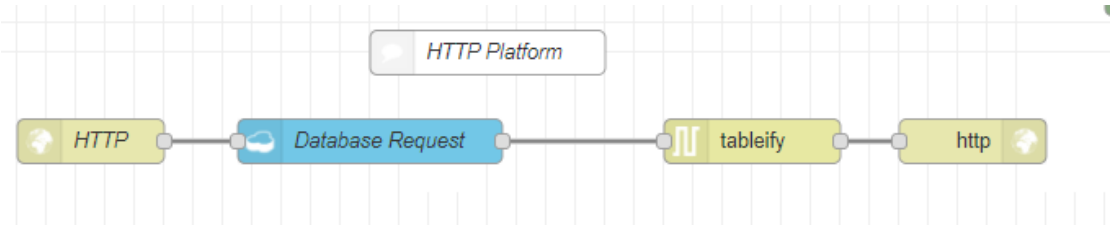

Figure 36 Flow to create a HTTP Platform of Database

SensorData		
Ambient Temp	payload	23.34
Object Temp	payload	18.81
Humidity	payload	12.65278
Pressure	payload	1025.94
Altitude	payload	20.64685
Acc X	payload	-0.98
Acc Y	payload	0.01
Acc Z	payload	0.04
Light	payload	14.01

Figure 37 Live Sensor Data in NodeRed Web Server

Figure 37 is a screenshot of web server data created in Figure 36 .

A real-time data visualising platform is also created by using IBM IoT Platform to further make this system more interactive. 6.1.4 contains some introductory contents about IBM IoT which will help us to understand this section more.

In Figure 38 a flow is created to send Sensor Data to IBM Watson IoT Platform.
But first we need to create a virtual device in IBM IoT by going into IBM Watson IoT platform through our Application Dashboard (Figure 29) and in add device we can create a device type and Once we create a device type we can register this to our Flow in NodeRed.

JSON function node in Figure 38 converts raw sensor data to JSON Objects which IBM IoT Platform understands to interpret data graphically. Function node's code can be accessed from Appendix 2.

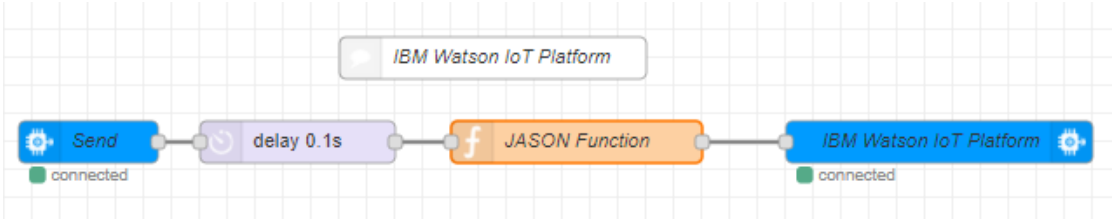

Figure 38 IBM Watson IoT Platform

The Device Type and Device Id (Figure 39) should be uniquely named which later needs to be registered to IBM Watson IoT receive node in NodeRed Figure 40.

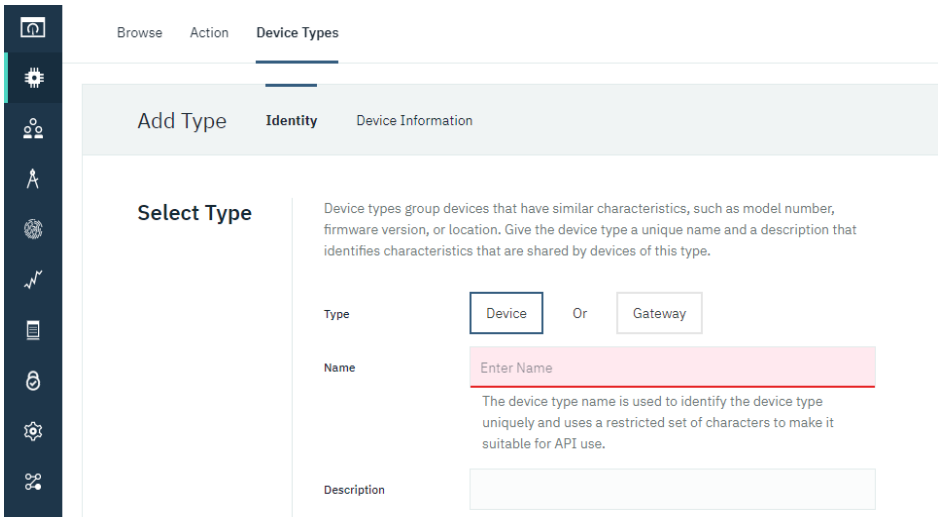

Figure 39 Defining Device Type in Watson IoT

Once we successfully establish a connection between NodeRed and IBM Watson IoT we can see the raw JSON data flowing (Figure 41). Now the mission is to create a good-looking dashboard to visualise these data.

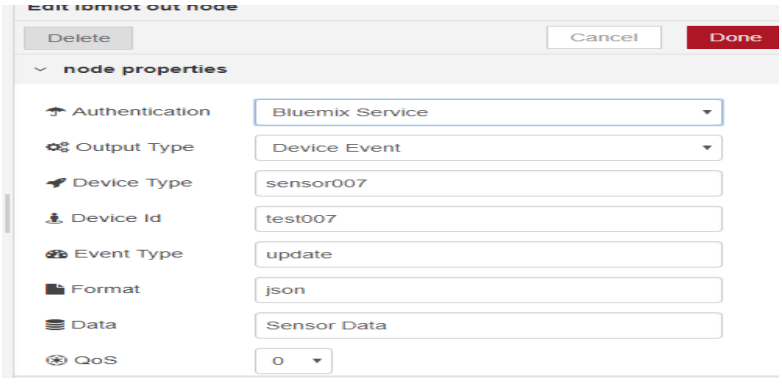

Figure 40 Defining Device Type and Device ID in IBM IOT out node

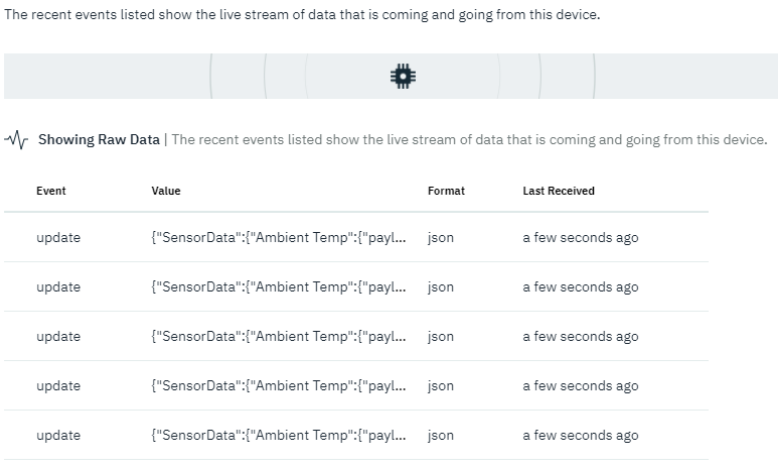

Figure 41 Raw JSON data coming from SensorTag to Watson IoT Platform

In Figure 42 inside the board's tab we can create a new board which has varieties of data visualising methods that we can choose from Charts, Bar Graphs and Line Graphs are few to mention.

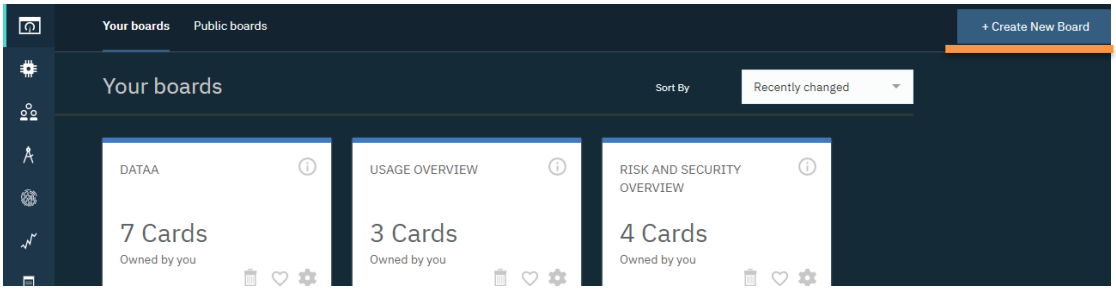

Figure 42 Creating Dashboard

Figure 43 is dashboard created to view live data from CC2650 SensorTag.

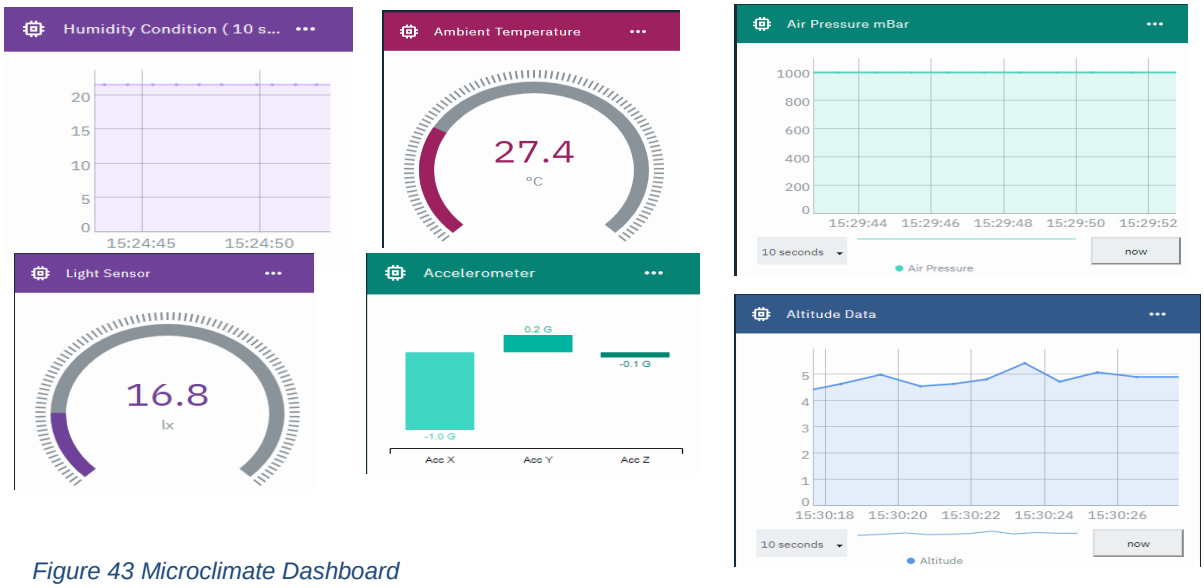

Figure 43 Microclimate Dashboard

9 Debugging

CC2650 SensorTag uses limited advertising feature to conserve battery power, after running for approximately 3 minutes and 180 seconds it goes to a lower power state and stops collecting data. Which in our case, is not suitable since we will be leaving our device in a remote location where human access could be limited. So SensorTag needed to be reprogrammed to advertise indefinitely.

Another hardware module called TI Debugger DevPack (Figure 46) was purchased to reprogram its firmware and disable the limited advertising feature.

9.1 TI Debugger DevPack

Plugging Debugger Figure 46 into a sensor tag adds debugging capabilities to SensorTag. Codes can be at once deployed and run into the microprocessor through a USB cable. SensorTag runs without battery when connected to the Debugger DevPack. With groove connectors other sensors and actuators can also be connected easily to expand its use [19].

9.2 BLE Stack

A BLE stack is a software development kit by Texas Instruments for low power devices, BLE Stack 2.2.2 is compatible with CC2650 SensorTag and can be reprogrammed with TI's Code Composer Studio to make device behavioural changes. [20]

BLE Device status of CC2650 SensorTag and current consumption is shown in Table 4

Table 4 Device Mode and Current Consumption [18]

Status	**Current Consumption**
Active (CPU is Running)	1.45mA + 31µA / MHz
Idle (CPU is not consuming power)	550 µA
Standby (CPU is not consuming voltage)	1µA
Shutdown	0.1 µA

BLE Stack 2.2.2 can be downloaded from Texas Instrument's Website: http://www.ti.com/tool/BLE-STACK.It contains BLE SDK Package which has SensorTag Source code and application to integrate other hardware with SensorTag such as an audio, light and LCD and other peripherals shown in Figure 44.

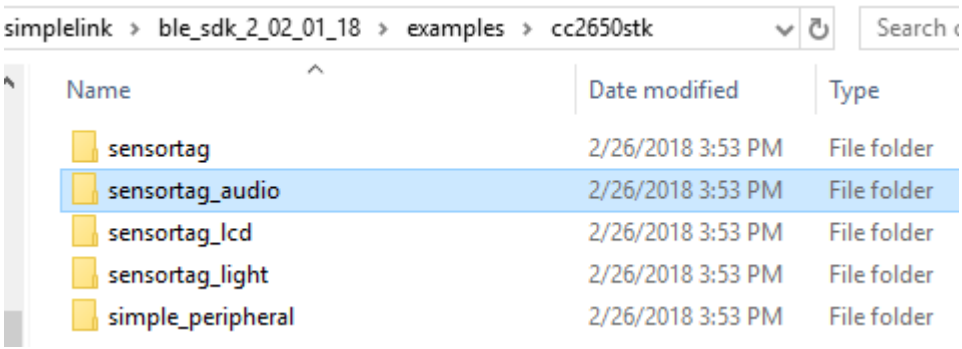

Figure 44 BLE SDK Contents

These files contents C program which can be opened from CCS directly.

9.3 Code Composer Studio

Code composer studio is a Texas Instrument's Integrated Development Environment (IDE) that supports DSP devices, microcontrollers and application processors.

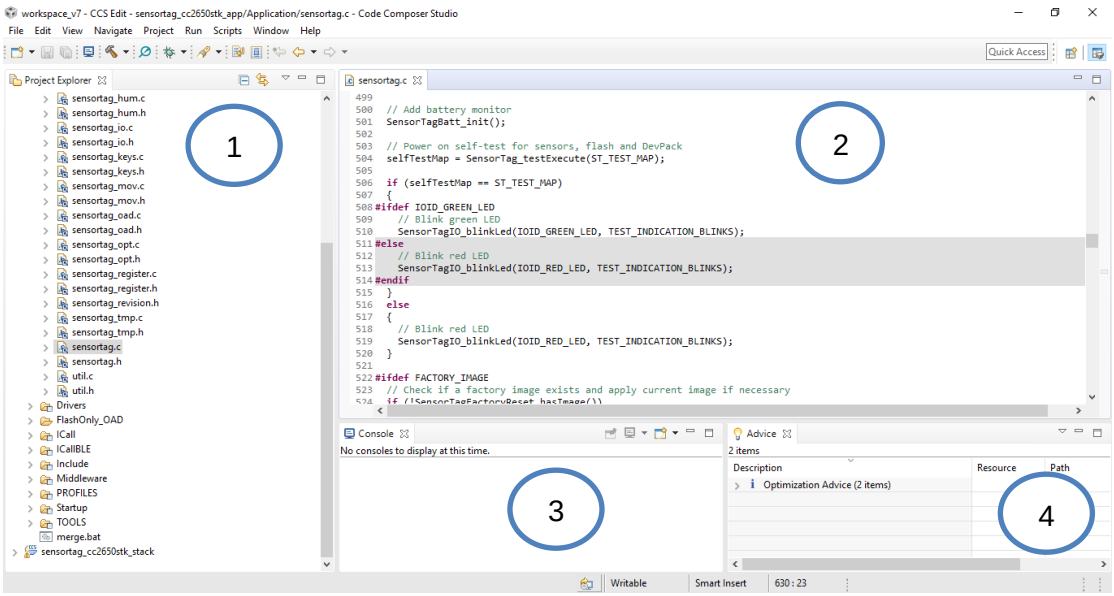

Figure 45 Code Composer Studio

Numbered bullets below are connected to Figure 45, Code Composer Studio window to introduce its general workflow.

1. Project View: Contains list of projects that is being created or extracted to be reprogrammed.
2. Text Editor: C/C++ files in project view can be accessed in Text Editor to make changes or new programs files can be written entirely.
3. Build View: Build View console displays the build steps when the code is executed.
4. Problem View: It displays build errors and optimization advices.

Once we get the general idea of Code Composer Studio we can plug in our SensorTag to Debugger Dev Pack as shown in Figure 46 .

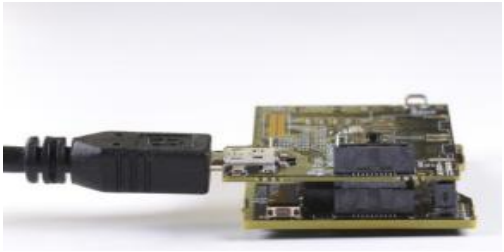

Figure 46 Debugger Dev Pack Plugged in with SensorTag [19]

While plugging the Debugger to SensorTag we should ensure that the SensorTag is not powered on. Once connected properly a green led blink on debugger when its ready to be used [19]. Once the debugger USB is connected, we can run code composer studio to begin.

To reprogram the SensorTag source code we need to be first familiar with BLE and BLE STK which are respectively described in section 5.6 and 9.2 of this document.

Once the CCS is running, we can extract the source code which is inside the folder SensorTag shown in Figure 44. When the file is extracted in project view window in Figure 45, we can see two projects *sensortag_cc2650stk_app* and *sensortag_cc2650stk_stack.*

Basic steps to follow to edit source code to advertise the SensorTag indefinitely can be accessed from Appendix 2 (Enable Advertising Protocol).

If no build errors were seen in the problem view window we can check *.hex files being generated.

Application *.hex file is under FlashOnly_OAD folder and Stack *.hex file is under FlashROM folder.

Once we have both Application and Stack hex file we need to merge it together and create a single. hex file, an IntelHex 2.1 python library needs to be downloaded to execute merge.bat file in CCS to combine app and stack.

But since we are directly connecting Debugger Dev Pack to SensorTag we can use TI Flash Programmer 2 to merge it and flash the hex file into SensorTag at once.

9.4 TI Flash Programmer 2

Once we have *.hex files ready we can connect the Debug DevPack with SensorTag as shown in Figure 46. Due to the risk of powering Debug DevPack by the battery in SensorTag it is recommended to use it with the USB cable attached. [19]

As the connection is ready, we can load our *.hex files previously created in 9.3 into Flash Images section in Figure 47 and run it to flash the SensorTag firmware.

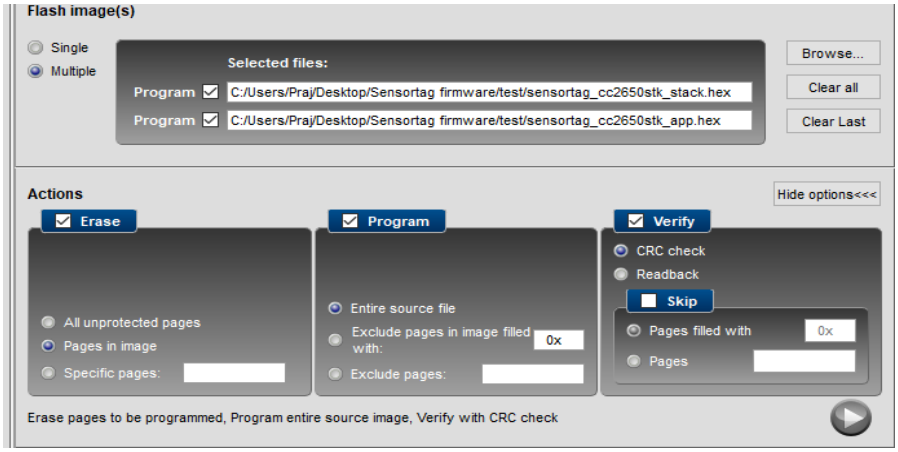

Figure 47 Flash Programmer 2

Once firmware is updated successfully green LED blinks rapidly which indicates the SensorTag is functionating normally. If something goes wrong and the device is not functioning as usual we can re flash the original firmware to fix the errors.

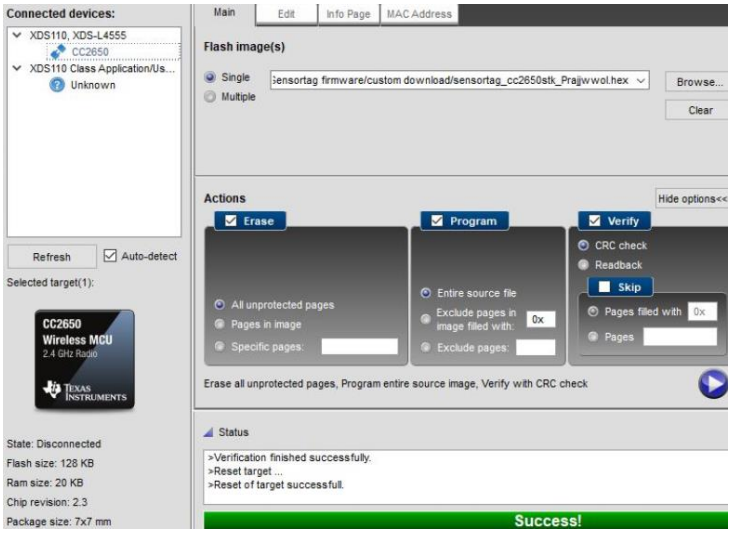

Figure 48 Flashing original firmware

The original firmware can be found in BLE Stack 2.2.2 folder which can be directly flashed from TI Flash Programmer 2 as shown in Figure 48, where original firmware, which is a single hex file is compiled again to reset the SensorTag to its original state.

10 Design Challenges

Setting up SensorTag was quicker than expected, downloading a mobile application and making a Bluetooth connection started data flow. Since our system needed to be working without any human presence just sending and receiving data on mobile was not enough and since we could only visualise data live, we needed a better system to store those data to make further future analysis.

IBM Quick Start cloud connectivity feature on the mobile application lead further study about the use of IBM Cloud to create a sensor system around it. IBM Cloud being a new topic and not many projects being done around it, it was bit confusing to understand IBM environment and its features. But the data sheets and guides around each application inside helped to understand it better to start the project. NodeRed was also a new programming tool. Although its relative straightforward, initially it took a while to

understand its workflow but a good documentation by the developers made things easier.

One of the most crucial tasks that were tackled for this project to be successful are:

- Limited Power Capacity
- SensorTag Advertising Issues

➢ Limited Power Capacity

CR2032 coin cell battery has a typical capacity of 240 mAh and can run for maximum of 48 hours with all sensors at work with maximum data sampling rate. With only few sensors working it can be powered for 240 hours [21] , which is not an ideal situation for our system so we had two alternatives to choose from one connecting a 3 V lithium ion battery to the soldering point which can be seen from Appendix 2 and Figure 51 and the second option which we decided to use is debugger and SensorTag, both connected to a USB Power Outlet since debugger can also be used to power up SensorTag also [19].

➢ SensorTag Advertising Issues

To save power SensorTag was programmed to switch off after few minutes of gateway inactivity [11]. As we needed our system to be working in remote environment and inside the bird nests, climbing up the nests to physically turn on the device was a big issue which we overcame by making few changes to the source code and updating the firmware which is described in detail in section 9.3 .

11 Tests and Results

Data collected over 30 min interval shown in Figure 49 shows that there have not been any issues with the device cloud platform and data collected throughout shows that the advertising issue of this device was solved and now can be used to collect data throughout time given the continuous power supply.

Collected data in Figure 49 can be accessed through this web link: https://microclimate.eu-gb.mybluemix.net/sensortag .

Figure 49 Data visualised in Web Platform

If SensorTag stays inactive for more than 10 days then IBM Cloud will automatically pause the application until application developer reactivates it again. So, when using pay per use service in IBM Cloud this feature will help developer to reduce runtime cost if there is a sudden hardware failure in the system.

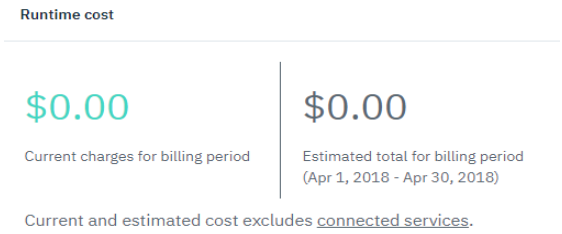

Figure 50 Runtime Cost evaluation

Runtime cost evaluated by IBM Cloud as shown in Figure 50 during the month of April with the currently running instances shows that with a 0-dollar investment cloud platform can be used with the microclimate module.

12 Conclusion

First part of this document discusses the general aspects of Sensor and Cloud Technology and its roles in IOT world, gives a little history of IOT and how it has grown till today and ways it will impact our future lifestyle. Sensors part includes general definition of sensor and briefly discuses some of the sensors that are used in SensorTag, Topic IOT talks about the history of IOT and examples of how it could be influencing our day to day lifestyle in near future. Then, an example in the context of Wildlife re-

search works done with sensor network technology is given and how earlier researches were done compared to today's development of sensor networks is discussed.

The middle part of the document solely focuses on understanding the design theory of microclimate module being developed, first it discuses about the basic technical aspects of SensorTag that is necessary to be understood before working with it then about understanding cloud computing in general and details about building a system with IBM Cloud.

A NodeRed programming guide can also be found among the topics that briefs about the concepts of NodeRed programming and why it is playing a crucial role in connecting devices in IOT world, the topic also describes how nodes and functions work inside the NodeRed environment and shows how a basic flow can be created to connect IOT devices.

The final part of this document discuses about creating full-fledged system with SensorTag and IBM Cloud, briefs about setting up the sensor device and creating an application with IBM Cloud and integrating it together to create the Microclimate Module. Ending of this report discuses design challenges that were tackled and ways those challenges were solved. Device advertising issues were successfully tackled by reprogramming its firmware to advertise indefinitely, issues with power was also taken care of by connecting the DevPack with SensorTag as DevPack connected to a USB power supply powered up SensorTag, which solved one of the possible disadvantages of this system, having to physically power up the SensorTag and replace the battery every once in a while.

Goal of this project was to design a cloud sensor module which will be able to read and extract various sensor data to study microclimate of nests. Based upon the tests and results, the goal of this project was met, a cloud sensor module was designed that can be used in nests for longer period without requiring any human interventions. Beside it's use for studying bird nests, this module can also be used for studying weather conditions of various places like laboratories and other spaces. Data Science students can use this system to generate a live weather data to practice on their data analysis courses, construction workers can use this device to track movements of their workers and other machineries. Its immense applicable possibilities make this module ideal door to enter the world of IOT.

13 References

[1] J. K. a. F. M. Hernann Rahn, “Microclimate of the nest and egg water oss of the eider Somateria mollissima and other waterfowl in Spitsbergen,” [Online]. Available: https://doi.org/10.3402/polar.v1i2.6982. [Accessed 25 3 2018].

[2] J. O. a. J. P. Carles Gomez, “Sensors,” *Overview and Evaluation of Bluetooth Low Energy:,* p. 20, 2012.

[3] M. L. Corbin, “developerWorks TV,” [Online]. Available: https://developer.ibm.com/tv/watson-iot-node-red-pi-dinosaur/. [Accessed 12 4 2018].

[4] K. L. Lueth, “IoT Basics : Getting Started with the Internet of Things,” [Online]. Available: https://iot-analytics.com/wp/wp-content/uploads/2015/03/2015-March-Whitepaper-IoT-basics-Getting-started-with-the-Internet-of-Things.pdf. [Accessed 1 4 2018].

[5] T. Bari, “A wireless sensor network to observe birds life,” [Online]. Available: http://www.es.ewi.tudelft.nl/msc-theses/2010-Bari.pdf.

[6] Weired.com, “The Ultimate on-the-fly Network,” [Online]. Available: https://www.wired.com/2003/12/network-2/. [Accessed 5 4 2018].

[7] T. Instruments, “Datasheet for CC2650 Sensortag,” [Online]. Available: http://www.ti.com/lit/ds/symlink/cc2650.pdf.

[8] Simplelink, “Datasheet for CC2650 SensorTag,” [Online]. Available: http://www.ti.com/lit/ds/symlink/cc2650.pdf.

[9] J. Teel, “Teardown of an Internet of Things (IoT) Wireless Device with Bluetooth Low-Energy and ZigBee,” [Online]. Available: https://predictabledesigns.com/tear-down-of-a-bluetooth-low-energy-ble-product/. [Accessed 16 3 2018].

[10] WIKIPEDIA, “Visible Spectrum,” [Online]. Available: https://en.wikipedia.org/wiki/Visible_spectrum. [Accessed 5 4 2018].

[11] T. Instruments, “CC2650 Datasheet,” [Online]. Available: http://www.ti.com/product/CC2650/datasheet/detailed_description#SWRS1587250.

[12] ARM University Relations, “ARM Cortex-M3,” [Online]. Available: https://www.arm.com/files/pdf/CortexM3_Uni_Intro.pdf. [Accessed 20 3 2018].

[13] Joseph Yiu, “The Definitive Guide to The ARM Cortex M3 (Secoend Edition),” [Online]. Available: https://www.eecs.umich.edu/courses/eecs373/labs/refs/M3%20Guide.pdf. [Accessed 21 3 2018].

[14] J. Decuir, “Bluetooth 4.0: Low Energy,” [Online]. Available: https://pdfs.semanticscholar.org/presentation/8d77/9b856fa9c0222480bced5e153eba7027c63e.pdf. [Accessed 2 4 2018].

[15] IBM Cloud, “Cloud Foundry App,” [Online]. Available: https://console.bluemix.net/catalog/starters/internet-of-things-platform-starter. [Accessed 22 3 2018].

[16] IBM, “What is Cloud Computing,” [Online]. Available: https://www.ibm.com/cloud/learn/what-is-cloud-computing. [Accessed 1 5 2018].

[17] J. Wende, “Node Red - A visual too for building Internet of Things,” [Online]. Available: http://www.iot-vienna.at/global-iot-day-event/2015/lib/exe/fetch.php?media=talks:wende_joerg:gide2015_node-red.pdf. [Accessed 22 3 2018].

[18] D. C. Jones, “Node Red Programming Guide,” [Online]. Available: http://noderedguide.com/. [Accessed 5 4 2018].

[19] Compel , “Debug DevPack User Guide,” [Online]. Available: https://www.compel.ru/item-pdf/10d01208ef0cac0ca5b064b9e6af3070/pn/ti~cc-devpack-debug.pdf. [Accessed 1 4 2018].

[20] C. L. a. M. H. Joakim Lindh, “Measuring Bluetooth Low Energy Power Consumption,” Texas Instruments, 2017.

[21] Misha, “Mobile Modding Tech Blog,” *TI SensorTag Power Consumption Analysis,* 2015.

[22] C. Jones, “w3.org,” [Online]. Available: https://www.w3.org/2014/02/wot/slides/conway-jones.pdf.

[23] Texas Instruments, “IOT Made Easy,” [Online]. Available: http://www.ti.com/ww/en/wireless_connectivity/sensortag/gettingStarted.html. [Accessed 25 03 2018].

[24] Postscapes, “Internet of Things Infographic , What Is The "Internet of Things"?,” [Online]. Available: https://www.postscapes.com/what-exactly-is-the-internet-of-things-infographic/. [Accessed 1 4 2018].

[25] Amazonaws, “IoT Harbor Postscapes Infographic,” [Online]. Available: https://s3.amazonaws.com/postscapes/IoT-Harbor-Postscapes-Infographic.pdf. [Accessed 5 4 2018].

[26] Smart and City, “Verizon IoT Smart Cities Seminar New York City,” [Online]. Available: http://www.smartandcity.com/event/verizon-iot-smart-cities-seminar-new-york-city/. [Accessed 30 4 2018].

[27] J. Lindh, “Connecting Machinery to the IoT,” [Online]. Available: https://www.ecnmag.com/blog/2014/12/connecting-machinery-iot. [Accessed 4 5 2018].

A. Bottom Layer of CC2650 SensorTag

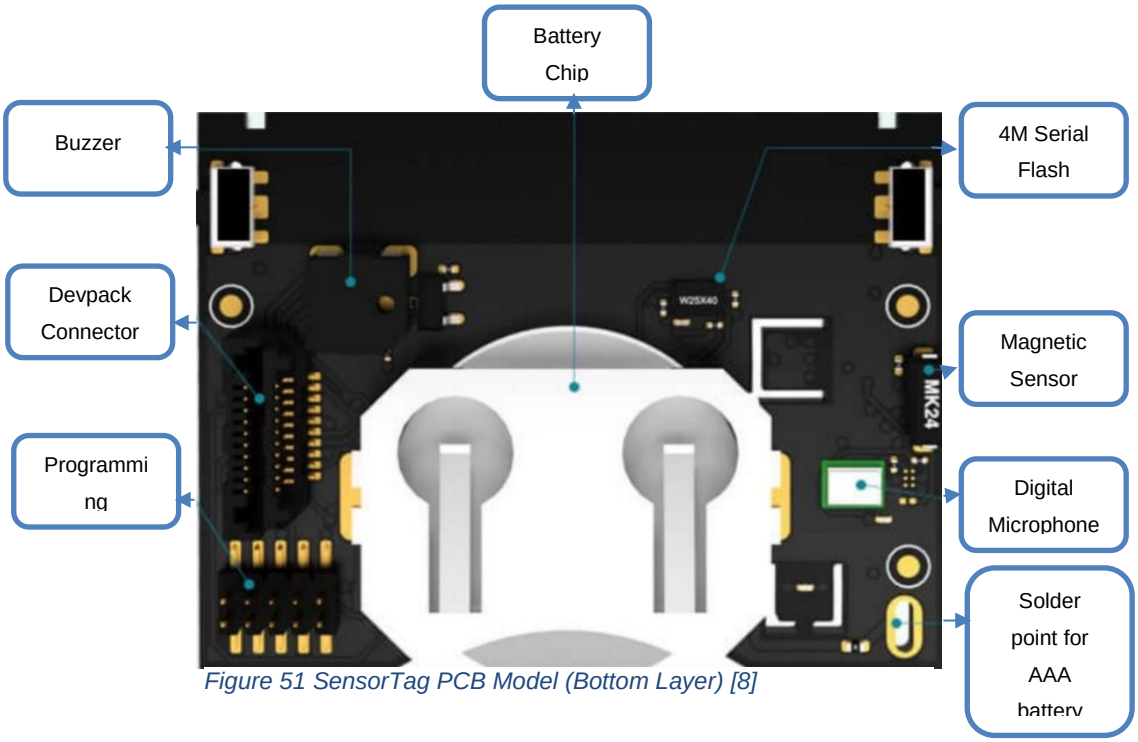

Figure 51 SensorTag PCB Model (Bottom Layer) [8]

B. SensorTag Configuration

Figure 52 SensorTag Getting Started Guide [23]

1. Download SensorTag mobile application to BLE Compliant device with Android (4.3+) or IOS

2. Connect battery with SensorTag by pulling the plastic switch.

3. When the SensorTag is advertising connect it with the mobile application.

A. Multiple Messaging Function in NodeRed [22]

```
var msg1 = { payload:msg.payload.d.ambientTemp };
var msg2 = { payload:msg.payload.d.objectTemp };
var msg3 = { payload:msg.payload.d.humidity};
var msg4 = { payload:msg.payload.d.pressure };
var msg5 = { payload:msg.payload.d.altitude };
var msg6 = { payload:msg.payload.d.accelX };
var msg7 = { payload:msg.payload.d.accelY };
var msg8 = { payload:msg.payload.d.accelZ };
var msg9 = { payload:msg.payload.d.light };

return [ msg1, msg2, msg3, msg4, msg5, msg6, msg7, msg8, msg9 ];
```

B. Indexing JSON Data with Respective Sensor ID

```
msg = {
  payload: JSON.stringify
(
        {
              d:{
                  "Ambient Temp" : msg1,
                    "Object Temp" : msg2,
                      "Humidity" : msg3,
                        "Pressure" : msg4,
                      "Altitude" : msg5,
                    "Acc X" : msg6,
                     "Acc Y" : msg7,
                        "Acc Z" : msg8,
                          "Light" : msg9,
                },
        }
  )
};
return msg;
```

C. Enable Advertising Protocol

Step 1:

Access: > *sensortag_cc2650stk_app*
> *Application*
> *Sensortag.c*

- Go to Line 114

```
            // General discoverable mode advertises indefinitely
#define DEFAULT_DISCOVERABLE_MODE             GAP_ADTYPE_FLAGS_LIMITED
```

Change LIMITED to GENERAL in Line 114

```
            // General discoverable mode advertises indefinitely
#define DEFAULT_DISCOVERABLE_MODE             GAP_ADTYPE_FLAGS_GENERAL
```

- Go to Line 430

SensorTag_init function

```
   // By setting this to zero, the device will go into the waiting state af-
ter
    // being discoverable for 30.72 second, and will not being advertising
again
    // until the enabler is set back to TRUE

            uint16_t advertOffTime = 0;
```

Change 0 to 1 in Line 430

```
            uint16_t advertOffTime = 1;
```

Step 2:

Now before building this project select:

> *sensortag_cc2650stk_app*

- Project
- Show build setting

> *Arm Hex Utility*

Enable *Arm Hex Utility*

Inside Show build setting > General > Select Compiler Version TI v5.2.6 for both projects.

Once we set the compiler we can build our project.

- Project
- Build Project